

Please
handle this volume
with care.

The University of Connecticut
Libraries, Storrs

WITHDRAWN



3 9153 01058227 0



Digitized by the Internet Archive
in 2023 with funding from
Kahle/Austin Foundation

The Research and Academic Users' Guide to the
IBM Personal Computer



The Lanyon Building, Queen's University, Belfast.

QA
76
.8
I25
R
19
V.

The Research and Academic Users' Guide to the IBM Personal Computer

VOLUME 2

Edited by
PAUL BARNETSON

IBM United Kingdom Limited

Oxford New York Tokyo
OXFORD UNIVERSITY PRESS

1988

Oxford University Press, Walton Street, Oxford OX2 6DP

Oxford New York Toronto

Delhi Bombay Calcutta Madras Karachi

Petaling Jaya Singapore Hong Kong Tokyo

Nairobi Dar es Salaam Cape Town

Melbourne Auckland

and associated companies in

Beirut Berlin Ibadan Nicosia

Oxford is a trade mark of Oxford University Press

Published in the United States

by Oxford University Press, New York

© IBM UK Ltd., 1988

This book is sold subject to the condition that it shall not, by way of trade or otherwise, be lent, re-sold, hired out, or otherwise circulated without the publisher's prior consent in any form of binding or cover other than that in which it is published and without a similar condition including this condition being imposed on the subsequent purchaser

British Library Cataloguing in Publication Data

The Research and academic users' guide to the IBM personal computer. Vol. 2

1. Research—Data processing 2. IBM Personal computer—Programming

I. Barnetson, Paul

001.4'028'54165 Q180.55.E4

ISBN 0-19-853744-1

Library of Congress Cataloging in Publication Data

The Research and academic users' guide to the IBM personal computer.

1. IBM Personal Computer. I. Barnetson, Paul.

QA76.8.I2594R47 1988 004.165 87-28134

ISBN 0-19-853744-1 (v. 2)

Set by Colset Pte Ltd, Singapore

Printed and bound

in Great Britain by Biddles Ltd

Guildford and Kings Lynn

Foreword

BRIAN WHITAKER

*Academic Systems Marketing Manager, IBM United Kingdom,
Ltd, Unit 125, Cambridge Science Park, Milton Road,
Cambridge, CB4 4FZ*

Welcome to the second volume of '*The Research and Academic Users' Guide to the IBM Personal Computer*'. It is designed for those of you in the academic and research community who use, or plan to use, an IBM Personal Computer.

Now that the IBM PC is established as both microcomputer and intelligent terminal, we felt that this second volume should concentrate on some of the different applications which British academics have found for the IBM PC. In this book you will find articles on using the IBM PC for many academic applications: from Classical Greek to communicating with other academics via JANET and EARN; from using Expert Systems in Biology to Phonetics Training; from Thermoluminescence Dating to Law; and many others.

Another area that has become significant in academia is the need to exchange information between different personal, or micro, computers, and articles in this book identify one of many such situations in which data and programs developed on one personal computer were needed to be used on an IBM PC.

Even if all these subjects are not precisely in your area, there are many tools and techniques described here which the authors have used, and which could be very valuable to you.

Now that we are into our second volume of *The Research and Academic Users' Guide* we are already looking forward to the next edition. If you would like to see particular topics covered in subsequent editions or you feel that you could provide useful

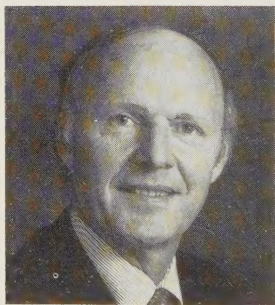
material for such a publication then please contact the editor:

Paul Barnetson, Academic Systems Marketing, IBM United Kingdom Ltd, PO Box 41, Northern Road, Portsmouth PO6 3AU

In particular, if there is significant reader feedback on moving data and programs between different personal computers, we will invite contributions which share similar experiences with other types of personal computer and the IBM PC, in future volumes.

I would like to remind you of IBM's support for the British academic and research community. IBM's Scientific Centres around the world are carrying out programmes of scientific research in collaboration with academic establishments. For example the IBM United Kingdom Scientific Centre in Winchester is currently working on image processing (particularly applied to medical problems), and speech synthesis. Elsewhere in the world, research is going on into areas as varied as knowledge based systems and environmental problems.

So here is the second volume of *'The Research and Academic Users' Guide to the IBM Personal Computer'*. We hope you find it useful. If you don't, tell us. If you do, tell your colleagues.



BRIAN WHITAKER MA MBCS

Brian Whitaker graduated from Gonville and Caius College, Cambridge, with an Engineering Degree in 1960. After completing a graduate apprenticeship with International Combustion Ltd of Derby, he spent five years in the motor industry, firstly as a Research Engineer and then as a Production Controller.

He joined IBM in 1965 as a trainee salesman. Since then he has held a variety of posts in Marketing, mainly with responsibilities for the academic community, and during this time he has had the opportunity to observe the developments that have taken place in academic computing. He is currently Manager of Academic Systems Marketing for IBM United Kingdom Ltd, and is particularly interested in the application of computers to the teaching process, in both secondary and tertiary education. He was also concerned in the formulation of IBM's academic initiatives as typified by the IBM Institute projects at Cambridge, Imperial College, and Manchester Business School.

He is a Council Member of the British Computer Society, and also sits on its Education and Training Committee.

Acknowledgements

ADA (or Ada) is the trademark of the Department of Defense, USA

Bugail is a trademark of the University of Bristol, UK

CAPTEX, the copyright of this package is owned by the Oxford University
Phonetics Laboratory, UK

Chitchat is a trademark of Sagesoft Ltd, UK

Clustan 2 is a trademark of Clustan Ltd, Edinburgh, UK

COSTAR was developed at the Massachusetts General Hospital, USA

Datamaster is a trademark of Software Solutions Inc., USA

dBase II and III are trademarks of Ashton Tate Inc., USA

Domino is a trademark of Compsoft Ltd, UK

DOSCOPY is available from BAKsoft, Cambridge, UK

Hasholme logboat, this photograph is reproduced by kind permission of
Hull Museum, UK

Kermit, the copyright of this package is owned by Columbia University,
USA, while the name is used by permission of Henson Associates Inc.,
NY, USA

LEXIS is a trademark of the Mead Corporation, USA

Lotus 1-2-3 is a trademark of the Lotus Development Corporation, USA

Micro-PROLOG belongs to Logic Programming Associates

Medical Query Language, this is the property of ICI and Computer Data
Services of Manchester, UK

Multiplan is a trademark of the Microsoft Corporation, USA

MUMPS was developed at the Massachusetts General Hospital, USA

PC-VT is a trademark of Mark C DiVecchio, California, USA

Peachtext is a trademark of Peachtree Software Inc., USA

pfs: File is a trademark of the Software Publishing Corporation, USA

Procomm is a trademark of PIL Software Systems, California, USA

Smarterm is a trademark of Persoft Inc., USA

STAF2 belongs to Calchem, USA

Tapestry is a trademark of Torus Systems Ltd, UK

Tekterm2 is a trademark of Greg Sherman, Virginia, USA

TeX belongs to the American Mathematical Society, USA

TL Emission from Crystals, the copyright of the photograph accompanying
from Mr Bailiff's article is owned by the Archaeology Department of
Durham University, UK

VIEW is a trademark of Acorn Computers Ltd, UK

WORD is a trademark of Microsoft Corporation, USA
WORDSTAR is a trademark of Micropro International

Every effort has been made to acknowledge all trademarks, together with the ownership of other items of software and hardware, which are mentioned in this book. If any acknowledgements have been inadvertently omitted or incorrectly attributed, IBM United Kingdom Limited would like to apologize, and if the error is brought to the attention of the editor it will be corrected in future editions.

Copyright

IBM United Kingdom Ltd authorizes all workers in the academic and research fields to copy, or otherwise use, parts of this book for research or educational purposes. An acknowledgement of the source would be appreciated.

Note

The opinions, statements, and findings, expressed in articles in this book are the responsibility of the authors of the articles, and their publication does not necessarily imply that IBM United Kingdom Limited agrees with them.

Remember that the art of writing good software is a swiftly advancing technology, and new packages for the IBM PC are being produced every day. So read comments in this book on specific packages with care, and note the date of both book and program.

Remember also that due to the inevitable delay between the work described here and the publication of this book, IBM workstations now available will be generally faster, cheaper, and more capacious than those used by the authors of these articles. The programs described here were written for, and run on, the IBM PC range that was current when the articles were prepared. It is IBM's understanding that very few, if any, changes will be needed to enable them to run on IBM's current range.



PAUL BARNETSON

Contents

List of contributors	xi
1. Law and the IBM PC	1
<i>C. M. Campbell and P. Leith</i>	
2. The IBM PC: a workstation for classical Greek	11
<i>D. H. A. Kaferly</i>	
3. Experiences with the IBM PC Network, and JANET/EARN access	19
<i>W. B. Dowsland</i>	
4. A dialogue processor on the IBM PC	28
<i>J. Whalley and D. Rush</i>	
5. Large-scale examination processing on an IBM PC AT	37
<i>P. Allatt, J. B. Slater, and M. Vassiloglou</i>	
6. Computer-assisted learning in chemistry, based on an IBM PC Network	42
<i>The CAL Group, Liverpool University</i>	
7. Developing expert systems for biological keys on the IBM PC	60
<i>J. Schofield</i>	
8. Using the IBM PC in the school classroom	71
<i>B. Millo</i>	
9. Logo and the IBM PC in action	80
<i>M. Thorne</i>	
10. Integrating the BBC Microcomputer and the IBM PC	89
<i>L. Burbridge and P. Buttner</i>	

11.	Moving programs from the Acorn BBC Micro to the IBM PC	97
	<i>A. Loyns</i>	
12.	Economic literacy and strategic planning for the executive—the role of the IBM PC	105
	<i>K. G. Lumsden and A. Scott</i>	
13.	An array processor for the IBM PC	112
	<i>R. E. Burger</i>	
14.	Computing and preventive cardiology with an IBM PC	122
	<i>R. W. Jones</i>	
15.	Phonetics training on the IBM PC	132
	<i>Anthony W. Bladon</i>	
16.	Thermoluminescence dating: an archaeological application of the IBM PC	138
	<i>I. K. Bailiff</i>	
17.	Sheep health, productivity, and the IBM PC	149
	<i>K. L. Morgan and A. T. A. Tuppen</i>	
18.	Using the IBM PC to simulate irrigation systems	156
	<i>M. A. Burton</i>	

Contributors

- Mr P. Allatt, *Computing Services, University of Salford, Salford, M5 4WT. Tel.: 061-736-5843*
- Mr I. K. Bailiff, *TL Laboratory, Department of Archaeology, Durham University, Fulling Mill, The Banks, Durham, DH1 3EB. Tel.: 091-3864466*
- Dr R. A. W. Bladon, *Phonetics Laboratory, University of Oxford, 41 Wellington Square, Oxford, OX1 2JF. Tel.: 0865-270440*
- Dr L. Burbridge, *Computer Centre, University of Bristol, University Walk, Bristol, BS8 1TW. Tel.: 0272-303343*
- Mr R. E. Burger, *Cambridge University Engineering Dept, Trumpington St., Cambridge, CB2 1PZ. Tel.: 0223-332765*
- Mr M. A. Burton, *Institute of Irrigation Studies, Southampton University, Southampton, SO9 5NH. Tel.: 0703-559122*
- Mr P. Buttner, *Computer Centre, University of Bristol, University Walk, Bristol 7, BS8 1TW. Tel.: 0272-303343*
- Professor C. M. Campbell, *Faculty of Law, Queen's University of Belfast, Belfast, BT7 1NN. Tel.: 0232-245133*
- Dr D. J. Chadwick, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Dr D. L. Cooper, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Dr W. B. Dowsland, *Department of Management Science and Statistics, University College of Swansea, Swansea, SA2 8PP. Tel.: 0792-205678*
- Dr P. P. Duce, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Dr R. W. Jones, *St Mary's Coronary Flow Trust, Waller Department of Cardiology, St Mary's Hospital, Praed St., London, W2 1NY. Tel.: 01-725-1014*
- Dr D. H. A. Kaferly, *Department of Greek, St Salvator's College, University of St Andrews, St Andrews, Fife, Scotland, KY16 9AL. Tel.: 0334-76161*
- Mr P. Leith, *Faculty of Law, Queen's University of Belfast, Belfast, BT7 1NN. Tel.: 0232-245133*

- Mr A. Loyns, *Department of Computer Science, The University, Manchester, M13 9PL. Tel.: 061-273-7121*
- Professor K. G. Lumsden, *The Esmee Fairbairn Research Centre, Heriot-Watt University, Chambers St., Edinburgh, EH1 1HX. Tel.: 031-225-8432*
- Dr D. Margerison, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Mr B. Millo, *Informatics Education Unit (HSTP), Dept of Education, University of Southampton, Southampton, Hants, SO9 5NH. Tel.: 0703-559122*
- Dr K. L. Morgan, *Department of Veterinary Medicine, University of Bristol, Langford House, Langford, Bristol, BS18 7DU. Tel.: 0934-852581*
- Dr D. Rush, *Department of Mathematics, Statistics, and Computing, Liverpool Polytechnic, Byrom Street, Liverpool, L3 3AF. Tel.: 051-207-3581*
- Mr J. Schofield, *Bedford College of Higher Education, Cauldwell St., Bedford, MK42 9AH. Tel.: 0234-45151*
- Dr A. Scott, *The Esmee Fairbairn Research Centre, Heriot-Watt University, Chambers St., Edinburgh, EH1 1HX. Tel.: 031-225-8432*
- *Dr J. B. Slater, *SWURCC, University of Bath, Claverton Down, Bath BA2 7AY. Tel.: 0225-60371*
- Dr M. Thorne, *Dept of Computing Mathematics, Mathematics Institute, University College, Senghennydd Rd, Cardiff, CF2 4AG. Tel.: 0222-874000*
- Mr A. T. A. Tuppen, *The Computer Centre, University of Bristol, Langford House, Langford, Bristol, BS18 7DU. Tel.: 0934-852581*
- Dr M. Vassiloglou, *Computing Services, University of Salford, Salford, M5 4WT. Tel.: 061-736-5843*
- Dr S. M. Walker, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Dr T. W. Walton, *School of Chemistry, Liverpool University, PO Box 147, Liverpool, L69 3BX. Tel.: 051-709-6022*
- Mr J. Whalley, *Department of Geology, Portsmouth Polytechnic, Portsmouth, PO1 3QL. Tel.: 0705-827681*
- Mr B. Whitaker, *Academic Systems Marketing Manager, IBM United Kingdom Ltd, Unit 125, Cambridge Science Park, Milton Road, Cambridge, CB4 4FZ. Tel.: 0223-861190*

1 Law and the IBM PC

C. M. CAMPBELL and P. LEITH

*Faculty of Law
Queen's University of Belfast
Belfast, BT7 1NN*

To a growing number of law schools the first contact with PCs has much in common with drowning. Unexpectedly, lawyers' professional training and impressive jargon seem to count for naught in the sea of computing which washes in. To the lawyers, who have long considered themselves more learned than lesser men, it is somewhat embarrassing to be unable to rise to the surface. What is more, there seems to be no escape—educational funders actually expect lawyers to pick up the relevant techniques and swim. But swim to where? That is the real problem at the moment—identifying the most useful applications for the available technology.

For a long time (since the 1960s, anyway) it has been axiomatic that computer-based legal retrieval systems are 'a good thing'. The legal databases are impressive, both in terms of size (LEXIS contains some millions of documents) and in terms of running costs (some £2 million per annum to update and service). But it is beginning to seem that the appeal is wearing a bit thin. For example, EUROLEX, one of the major legal databases servicing the UK, recently went out of business, and LEXIS staff constantly point out to academic users (forever requesting more discounts) that LEXIS needs proper revenue to continue. With this foundational stone now rather shaky, might it be that the very idea of applying computers to the law is mistaken? So far the uses of computers in the legal profession are mundane: word processing, time-recording, and accounting. Can computers offer genuine benefits to the art of 'lawyering' itself? Time will tell—but only if researchers try to make a success in combining law and computing.

In this article we point to some of the efforts to harness computers to legal work; we speak from practical experience since in the Faculty of Law at Queen's University we are actively involved in most of the

relevant areas. And for those who might think that computing has only low-level practical utilities to offer lawyers, we point out that the very act of applying computers to law poses important questions about the legal and legislative process. For example, is the law simply a collection of rules which can be held in a computer program, or is it something 'beyond rules' which the computer will never be able to encapsulate? Is it possible to conceive of a legal *system* of internally consistent and entirely coherent precepts and norms? Similar questions have been asked for centuries by philosophers of law; with computing equipment being applied increasingly in law schools, the questions are becoming more relevant than ever.

Breaking away from the mainframe

One of the first points of interest is why law schools seem to be concentrating on microcomputers. Why are they not using the mainframe which has been available in the campus for years? To computer scientists this might not be immediately obvious, since they themselves are the most fêted users in a university or polytechnic environment. One reason is that most law schools have been ignored in the planning of communications—they have no communications lines which allow them to connect easily into the 'common' computer facilities of their institution. This actually reflects a wider point—previously computers and technology were not 'for' lawyers. Like other disciplines in the humanities, law was funded and structured to use libraries and books but had no need of expensive equipment. A second reason is the complexity of most mainframe logins. It usually takes a substantial amount of individual supervision to help the law student through the initial problems of connecting to, and then logging onto the mainframe. This is not a trivial problem—law schools rarely have computer advisors or staff to help 100 first-year students through this psychologically baffling period. Even after login, the mainframe typically lacks the user-friendliness lawyers need. Self-booting IBM PC software overcomes the login difficulty, and many programs now available are easy to use. A third reason is that most software which is of interest to law schools is appearing in IBM PC format—and since computing hardware's *raison d'être* is to run software, legal educators are taking the sensible hardware solution. At a recent meeting of law schools, a straw poll suggested that the IBM PC format is becoming the *de facto* standard. It is likely that

more pressure will arise to accord with this standard, since teaching software for law is currently at a premium.

We should add that there is also a well-founded view amongst those interested in legal computing that, somehow, computer centres do not understand their needs as they understand the needs of the Fortran programmer or the SPSS user. This has led to a feeling that—given sufficient hardware—law schools can, and have to, do it themselves. In a way, law schools now are in transition—coming from a past when computers were not for them (they were for engineers and scientists) to a future when computers will be relied on for many aspects of legal work.

Teaching legal computing and law

Law schools train their students in the provisions and precepts of the law. Students are taught to understand or search out relevant legal rules and principles to solve legal problems, advise on individuals' rights, etc. They are rarely, except in courses on legal computing, taught to look on law as a profit-making exercise. Nor are they often, except in courses on legal computing, taught to look upon the law as an information-handling process. And yet, when they do move into practice, such day-to-day concerns will occupy them much more than 'legal research' ever will.

We began to introduce undergraduate students to computers in 1974 and since 1978 have offered a specialist optional course in Computers and Law. In the current course, students are expected to have practical sessions on IBM PCs. We have our own computer laboratory. We teach them little of the programming of the machines, instead supplying them with a floppy disk, a list of instructions to follow and a computer to use. One of the practicals deals with the preparation of legal documents, an aspect of legal work which will fully occupy them in practice, and which, some people suggest, will be the subject of much development in the near future.

Briefly, the student is led through the practical, starting with a simple document preparation system. This program reads a 'blank' document form (called a 'precedent') and then prompts the user for input (such as name of vendor, date, etc.) which can be included in the blank document and then printed out. By making the program more complex (reducing the number of questions by re-using input, adding simple validation routines, and making use of printer control

codes) the student is led through various aspects of the design of 'user-friendly' office systems in a much more instructive way than simply reading about it. It is surprising how some students become involved with such a simple set of programs—they see, sometimes for the first time, how computers can be of real benefit in legal practice, and how badly designed systems can upset potential users.

Some educators believe that teaching law students about actual legal practice is something best left until after they receive their degree. But, it seems to us that simple practical sessions with well-designed teaching software can do so much to heighten the student's awareness of the legal process. Thus, a teaching program which demonstrates how the (increasingly computerized) Magistrates Courts operate, can be useful to the student. First, it indicates how 99 per cent of legal actions can be understood as primarily the clerical handling of documents. Secondly, court processes can be viewed as a sociological phenomenon with little or none of the legal pedantry which academic instruction suggested the practice of the law to be. And, thirdly, there is insight into the day-to-day running of the legal process—where the student's future time will be spent. To the law school with available hardware, the design of such teaching programs is one way in which they can integrate computing with the wider aspects of legal education.

Computer-aided learning in law

Whereas a course on legal computing is most interested in getting the students familiar with computing and how it may be applied in the legal field, CAL is the application of computers to law where the student should have little interest in the computer—only in the law. It is becoming an area of some respectability; even Harvard University has a project to prepare teaching material in this form!

British academics, it is said, are touchy about being observed teaching, yet expect their examining of students to be open to scrutiny. American academics are said to be nearly opposite—to have little unease about being observed in class teaching, yet sensitive about their examination practices. This might explain why legal educators in America took the lead in CAL work—they saw it as being an extension to their already 'public' teaching.

Few authors of legal CAL programs suggest that CAL can supplant the role of the lecturer. But there are large areas of any subject which can be aided by sensitive use of CAL. Thus, one

academic from the commercial law area in the USA wrote 'For instance, for a piece of paper to qualify as a check or promissory note there are certain technical requirements—a certain magical language that must be used. And that is perfect for a computer-aided exercise.'

Legal CAL programs can be written to deal with 'drill' exercises, 'tutorials', and 'simulation', each application targeting on a different aspect of law teaching. Drill exercises are the simplest, and the simplest to program; they are composed of a series of short questions which expect a 'right' or 'wrong' answer—they are useful in reinforcing understanding of the components of legal provisions. Thus, say that the author of a program wished to deal with 'Mareva Injunctions' (these are remedies which can be used in certain situations) the CAL program might have a list of questions which ask, 'Is the Mareva Injunction suitable for the case where . . .'. By immediately telling the students whether their answers are correct or incorrect, a means of simple reinforcement can be achieved. Of course, legal teachers would raise their hands in horror if anyone suggested this was all there was to legal education. But drill exercises do have a role to play—they help the student to think about why their answers were wrong, and to that extent can encourage individual development.

More sophisticated is the 'tutorial' CAL program. The sophistication comes from allowing 'free form' input of answers, and there is usually a more complex correcting of the student's errors—by giving the errant student more questions which are designed to clarify the particular concept, and letting the correct student go to the next section. The free-form input has been found to be satisfactory, because so much of the law requires mention of specific points or concepts—an advantage which arises from having a well-developed legal jargon. It is thus possible to ascertain whether an answer is correct, say, if it has the word 'hearsay' or 'alibi' in it.

The third kind of CAL program attempts to simulate real life situations—court-room behaviour and suchlike, or uses game formats for two or more students to compete. This third sort is most suitable for linking up a video-disk to the IBM PC.

At Queen's we are concerned with the first two CAL applications, and are specifically interested in the design of IBM PC-based CAL-authoring languages for the law. All other presently used languages have been purloined from other areas, such as chemistry (with STAF2). One initial system was developed in our faculty to allow a

CAL program to make direct use of a database, the student being able to switch easily between the database which holds complex government regulations and the CAL program which tested practical use of these regulations. We intend to develop this idea more fully since law is such a text-based subject; if the student has access to up-to-date machine-readable materials on the IBM PC, then CAL ought to be even more efficient.

And it is not just current legal provisions which can be used on a combined database/CAL system. One lecturer at Queen's is planning to use the texts of a nineteenth century legal philosopher as the content of the database. This will—he feels—improve the student's reading of the primary texts of his course.

Electronic publishing

Not so long ago, printing was an occupation involving heavy beds of typesetting and much hot metal. Only in recent times, with the advent of computerized typesetting, has it become something which could be done on a computer screen and a phototypesetter. Now it is something which can be successfully carried out on a hard-disk-based IBM PC with a laser printer.

Most law schools keep close contact with the practising members of the profession—whether solicitors or barristers—and they are often seen by members of the profession as a source of up-to-date legal advice. For example, some of the smaller law schools have members with particular interests in, say, the application of law to the new technologies and it could be that these academics might wish to produce a newsletter. Very high quality and professional camera-ready copy can now be produced with a limited IBM PC-based system—a distinct improvement on newsletters which depend on the university's outdated printing facilities.

There are also larger publishing ventures within Law Faculties. At Queen's—as the only law school serving the Northern Irish jurisdiction—we have what is the equivalent of an in-house publishing company (SLS, standing for 'Servicing the Legal System') which produces a large number of different kinds of materials for the legal profession in Northern Ireland as well as for academic use. Thus, textbooks on N.I. law and current awareness journals are produced by SLS. The high cost of typesetting this material commercially suggested that it might be possible to produce much of it as camera-ready copy in-house, especially since most of

the future output will be prepared by members of the faculty on IBM word-processing equipment. Some work has been carried out on the design of such a system, and we hope to have a working demonstration system arranged in the near future which can be used to edit and typeset complex legal material in as 'automatic' a manner as possible.

The advent of micro-publishing has come from the existence of pieces of software, such as TeX which was designed by Donald Knuth to handle complex mathematical typesetting. At one time it was necessary to have a large mainframe to run this software, but now it can be handled on relatively small IBM PC XT systems with laser printers costing—in total—less than £10 000.

Each law school has its own publishing problems, and it is the flexibility of the typesetting software which is most important. The school can set up a system (perhaps with the help of their Computer Centre) which can deal with their own problems; frequently the cost-effectiveness of the legal publications will be supported by 'administrative' needs, e.g. printing exam papers.

Decision support

The computer application which is currently at the research pitface of law is that dealing with 'expert', IKBS ('Intelligent Knowledge-Based Systems'), or decision support systems as they are more modestly termed. Such systems will constitute a step forward from the (relatively) simple information retrieval programs which have been available for so long. The problems with an information retrieval system in law are that, once the material has been found, it still has to be interpreted and understood within the light of a large number of other legal documents. Also, one cannot ask such a retrieval system a direct 'legal' question—one can only search for words (any string of characters) which might occur in discussions of the legal question. It is these problems which support systems are to overcome. They ought to be able to help with direct questions, and they also ought to provide enough information to the users so that they can readily make sense of the legal position pointed to by the retrieved text. Much, if not most, of the current research in IKBS is involved with the building of rule-based programs. These programs have managed to provide a high level of expertise in discrete areas in, for example, medicine and the physical sciences, but the rule-based approach has not yet proven itself in the legal field. We use the term

'decision support' to describe the sort of systems we wish to design because, first, we want to get away from the heavy rule-based emphasis of current IKBS; and, secondly, because we are aware that any system for the legal profession must give advice and aid (for the many and varied tasks carried out by the lawyer) rather than simply presenting a legal rule to follow. Thus, our system should be designed to support the decisions of the lawyer, rather than to present him with a rule to be followed mechanistically.

At Queen's we are planning something more sophisticated than a pure rule-based system. We are working with the idea that it ought to be a program which builds on the techniques developed in traditional information retrieval, but has more sensitive and extensive indexing, searching, and presentation features. Whether our approach will be successful is something that only the future can tell.

Conclusion

The problems which can be met when a number of IBM PCs are delivered to a law school should not be underestimated. Despite their small size and low cost, they are complex pieces of equipment which can either enhance teaching and research in a faculty, or can accumulate dust in a little-opened cupboard. One of the first points which occurs to most legal academics is that they can be used for word processing. This is, of course, quite true—both students and their teachers can use them in the preparation of articles and research papers. Keyboard literacy and word-processing skills will be a normal part of office life for most people before long. But it would be sad if this was all the computers were to be used for. Even using the word-processed text as input to an in-house publishing venture is a positive step—and one which can be taken for little extra expenditure in software and hardware.

It is also the case that the computer can be used to develop the student's understanding of the law in both the practical and the theoretical sense. It would be a valuable contribution to legal education if schools kept this in view at all times.

At Queen's we are luckier than most law schools. We have all the hardware we need—provided by the University or secured as part of research contracts. We also have a complement of two programmers (one working on database systems, and one on CAL research) together with a member of faculty who is a computer scientist. It would be the ideal solution if all law schools were able to access such

in-house expertise in computing, rather than having to go, cap in hand, to their computer centres.



COLIN CAMPBELL

Colin Campbell taught in the Universities of Dundee and Edinburgh before becoming Professor of Jurisprudence in Queen's University in 1974. Having taught law students something about computers in connection with privacy issues from 1967, he began teaching computer applications to law in undergraduate courses in 1974. He was a founding member of the Scottish Legal Computer Research and of the Society for Computers and Law, and has written various articles and books in the field of technology and law: he was co-author of *Computers for Lawyers* (1971), and editor of *Data Processing and the Law* (1984). He is a former Dean of the Faculty of Law in Queen's and is now Senior Pro-Vice Chancellor of the University.

PHILIP LEITH



Philip Leith gained a B.Sc. in sociology from Edinburgh University in 1976, a Dip. S.A.D. (Systems Analysis) from Napier College in 1980, and his Ph. D. from the Open University in 1986. He began his research into the application of computers to law in 1981, using rule-based techniques. On leaving his lecturing post at the Open University in 1985 he moved to a New

Blood post in IT and Law at Queen's, where there was a long history of active research into computers and law. Much of his recent personal research has been spent developing an understanding of why current interactive knowledge-based systems or logic programming approaches are not suitable for the legal area, and considering other possible approaches to the lawyer's need for expert computerized information.

2 The IBM PC: a workstation for classical Greek

D. H. A. KAFLERLY

*Department of Greek
St. Salvator's College
University of St. Andrews
St. Andrews
Fife
Scotland, KY16 9AL*

The present past

What is a classicist doing in a publication like this? Classical Greek, statistics, and computers are traditionally mutually exclusive areas of research. What brought about the change? In the beginning there was χ , now, there is χ^2 . In the fullness of time I must compute the latter in order to understand the phonological usage of the former. This is possible *if* and *when* I take classical Greek not only to be literature and letters, but also to be machine texts and data. Implicit in that 'discovery' is the recognition that a computer bridges the gap between Greek and statistics. Furthermore, it is clear that what was first possible only on large mainframe systems may now be transferred and executed entirely on an IBM Personal Computer.

Is it as easy as that? Does research in classical Greek, and arts in general, lend itself to the vigorous and systematic constraints imposed by both statistics and computer programs? In a recent paper, a statistician points out:

...
an arts researcher does what all perceptive readers do—reads a text until he is struck by a recurrent rhythm, phrase, or some other striking effect. Only then does he use the methods of the statistician to confirm or refute the hypothesis that the target author consistently uses this intuitively isolated feature with greater frequency than comparable writers. (Holmes 1985.)

Statistics, at best, may suggest probabilistic tendencies in

literature. It does not offer proofs. The most potent tool for stylistic evaluation remains intuition.

In my principal area of research, fifth century (BC) drama, I did not even need intuition for I took up one of classical Greek's oldest and most notorious feuds, the charge of Euripides' excessive sigmatism. Sigma, the letter 's', was adversely regarded by literary critics in antiquity, some of whom dubbed Euripides, 'sigma-loving'. This was a useful link between studies of quality and those of quantity. I had the surviving plays of Aristophanes, Euripides, and Sophocles (36 in all) in machine readable form (approximately 3 Mbytes). However, in order to fully understand the contribution of sigma as a 'sound' in their works, I first needed detailed information concerning the distribution of all the 24 letters in the 36 plays. It would not be enough to know just *how many* sigmas were contained in an author's play but *where were they*?

Method: classifications and categories

To this end each play was treated as a pool of letters, the letter count was refined in terms of vowels and consonants and the six consonant groups: gutturals, labials, dentals, liquids, aspirates, and sibilants. The frequency of every letter in an initial, medial, or final position within a phonetic word and within a verse line was recorded. Each play, and subsequently each author, was described in terms of vowel : consonant ratio, consonant group representation, consonant group position (initial, medial, or final), and consonant group alliteration in trimeter scenes.

The collected letters of Aristophanes, Euripides, and Sophocles

There were over 1 500 000 letters in the 36 plays. Data management, storage, and retrieval was of paramount importance. This required planning ahead, anticipating the types of statistical tests and graphics packages to be used. Before I could make any inferential judgements from the data I had to employ general, descriptive statistical analysis to determine whether the variance measured within a play was larger or smaller than that between plays. That is, I had to show that what was being measured varied more between the works of the three authors than that measured among different plays of the same author. Now, remembering that statistics cannot prove,

but only merely suggest probabilistic tendencies, the results at that level of analysis *strongly* suggest that there exists three such distinct probabilistic tendencies in literature which we may term Aristophanes, Euripides, and Sophocles. Unique 'voice-prints' for each author emerged which indicated individual traits, preferences, and some time-dependent features. Differences between tragedy and comedy were measured and the charge of Euripides' excessive sigmatism was proven.

The collected letters of Sophocles reveal a vowel : consonant ratio (VCR) which favours the consonants (Fig. 2.1) but which also indicates a slight increase in the proportion of vowels with time. Trimeter sections in Sophocles' plays are characterized by low vowels and high consonants. This inverse relation is reflected in the configuration of the phonetic word. Trimeter sections contain low initial vowels and high initial consonants, whereas the lyric sections contain high initial vowels and low initial consonants.

48.44	vowels	48.22
5.90	gutturals	6.08
5.00	labials	4.86
11.47	dentals	11.71
17.45	liquids	17.20
4.37	aspirates	4.32
7.38	sibilants	7.61

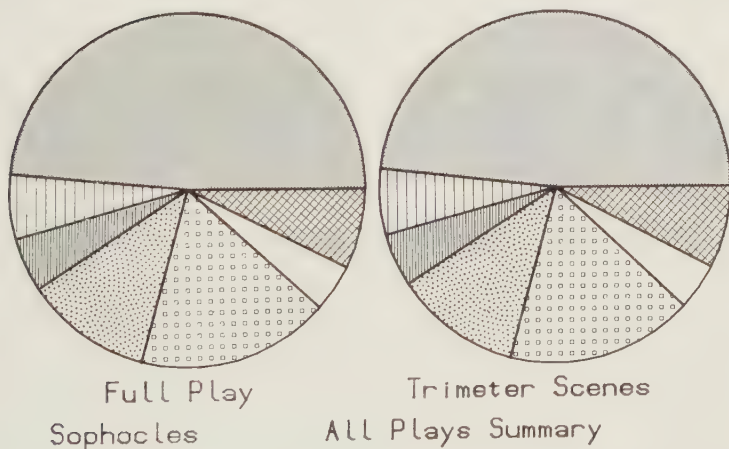


Fig. 2.1 Proportions of vowels and the six consonant groups in the full play and trimeter scenes of Sophocles' plays. Using data from the summary of all plays.

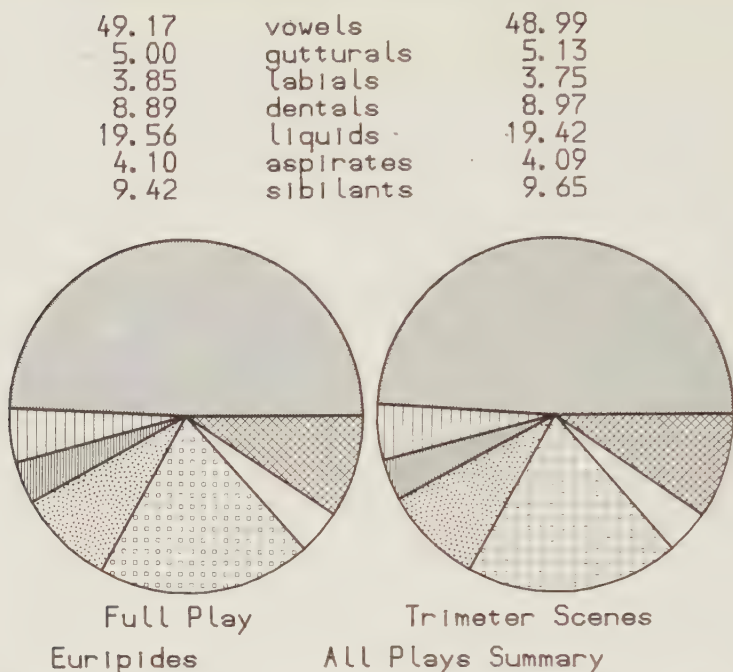


Fig. 2.2 Proportions of vowels and the six consonant groups in the full play and trimeter scenes of Euripides' plays. Using data from the summary of all plays.

Data from Euripides' collected letters agrees with that of Sophocles in terms of proportional representation, but differs from Sophocles in terms of degree (Fig. 2.2). That is, Euripides was found to be more consonantal than Sophocles in every category.

The collected letters of Aristophanes, on the other hand, show a distinct and consistent preference for vowels over the consonants (Fig. 2.3). In contrast to Sophocles and Euripides, Aristophanes' trimeter sections are marked by high vowels and low consonants, and his corresponding lyric sections by low vowels and high consonants. In a broad sense, this may mean only that one out of the three authors examined, employed proportionally more vowels than consonants. However, the fact that Aristophanes' plays present a different dramatic form from the other two authors lends his data added interest. The possible significance of this difference was examined by reviewing the distribution of vowels and consonants

50.33	vowels	50.56
5.27	gutturals	5.36
4.15	labials	4.13
9.87	dentals	10.09
18.96	liquids	18.79
3.41	aspirates	3.35
8.02	sibilants	7.73

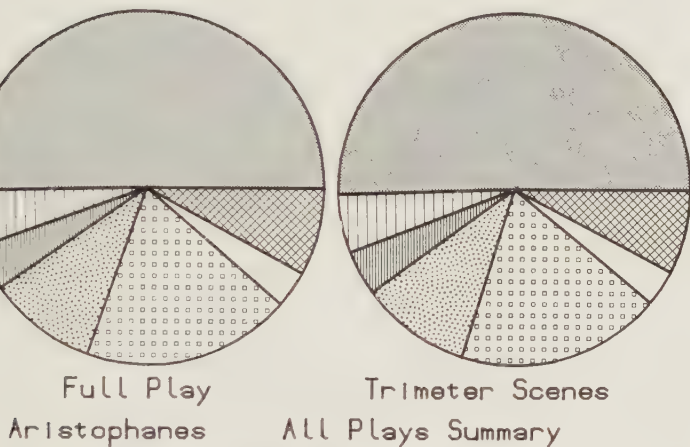


Fig. 2.3 Proportions of vowels and the six consonant groups in the full play and trimeter scenes of Aristophanes' plays. Using data from the summary of all plays.

within the full play, the trimeter sections, and the lyric portions. For, if consonants are connected with tragedy, then how and where do they appear in comedy?

Tragedy and comedy: a measured contrast

In general, tragic trimeters were found to contain proportionally more consonants and fewer vowels than their corresponding lyric sections. However, comic trimeters contain proportionally more vowels and fewer consonants than their lyric sections. Tragic lyrics are characterized by a greater proportion of vowels whereas comic lyrics contain a greater proportion of consonants. In sum, tragic trimeters are consonantal (closed) and its lyric sections vocalic (open). Comic trimeters are vocalic (open) and its lyric sections consonantal (closed). This suggests that in the fifth century BC,

tragedy and comedy, the two faces of drama, were phonetic mirror-images.

There are, no doubt, many complex explanations for such a measured effect. The difference, in fact, may be simply the natural contrast in sounds between that which is spoken and that which is sung. But, if this were only the case then the differences between spoken and sung sections should transcend dramatic form. That is, comic lyric sections should be vocalic, too. The denied expectancy, one 'expects' vowels but 'hears' consonants, may be one radical level of comic parody of the tragic form.

Sibilant rivalry

At every level of the review, from crude letter count to trimeter alliteration, Euripides' proportional representation of sigmas and, therefore, of sibilants always exceeded those of Aristophanes (sometimes by as much as 3 per cent) and Sophocles (by as much as 1 per cent). In the main, Euripides' plays contain proportionally more sigmas, which are found in consistently greater representation in one part of the play (trimeter sections) and, within that part, frequently occupy a specific position within the phonetic word (initial), thus providing the conditions for significant sibilant alliteration within a trimeter line.

In this way, and through the computer's 'ear', I could 'hear' Euripides and account him highly sigmatic. This empirical evidence corroborates the judgement of his peers. The computer also detected and isolated sigmatic passages in Sophocles and Aristophanes. In every case, however, the occurrence may be defended in terms of literary convention, for example a dramatic character *in extremis*, or by localized effect, a comic chorus performing a tragic parody. On the other hand, Euripides is sigmatic not only *in extremis* but *in general*. Thus, conventional and excessive sigmatism are fundamental characteristics of Euripides' plays.

Computer dating

One advantage of collecting and organizing data into a general database is that particular nuggets may filter out. In spite of looking for sigma data, other issues emerged. One such concerned an apparent shift towards more initial consonants in the trimeter sections of Sophocles' later plays. Sophocles produced over 100

plays, of which seven still exist. Assessment of his development, however, is hampered by uncertainties of chronology. Only *Oedipus Coloneus* (401 BC) and *Philoctetes* (409 BC) are securely dated. For the remaining five plays there is some general agreement on a loose, relative chronology, but, of these, *Trachiniae* is the most problematical and hard to place. The traditional ordering is *Ajax*, *Antigone*, *Trachiniae*, *Oedipus Tyrannus*, *Electra*, *Philoctetes*, and *Oedipus Coloneus*. But, if the proportion of initial consonants is indeed found to be a time-dependent feature, then *Trachiniae*'s position after *Antigone* and, for that matter, after *Ajax* becomes suspect.

Using CLUSTAN 2, a statistical package developed at the University of St Andrews and updated for the IBM PC, I was able to apply principal component analysis to determine a linear combination of characteristics and to find that two-dimensional scatter diagram which would best describe the relationship among the seven plays. Each play was represented by a point in six-dimensional space, one for each of the consonant groups. The distance between the points corresponds to the similarities between the plays. In this way I obtained a two-dimensional diagram which suggests that *Trachiniae* belongs to an earlier period of composition and production. With further research and testing, the order of Sophocles' plays may prove to be: *Trachiniae*, *Antigone*, *Ajax*, *Oedipus Tyrannus*, *Electra*, *Philoctetes*, and *Oedipus Coloneus*.

The future perfect

I have reviewed the IBM PC as a private workstation. The IBM PC will function as a public facility as well. It is self-sufficient to my immediate research requirements due to its inherent strength, flexibility, and independence. Yet those qualities plus software portability and operating system compatibility combine to support maximum public contact via Britain's Joint Academic Network (JANET), North America's BITNET, and by satellite to Australia. In this way and in the near future, a researcher in classical Greek will be able to build programs, execute or modify models, collect and analyse data, etc., without concern for where the tools, programs, or models reside, all on their local IBM PC workstation. Eureka!

Reference

Holmes, D.I. (1985). The analysis of literary style—a review. *Journal of the Royal Statistical Society A***148**, (4), 328–41.



DIANE KAFERLY

Diane Kaferly was the first IBM Research Fellow to be appointed in the Department of Greek at the University of St Andrews. Before this she was formerly on the technical staff of Bell Telephone Laboratory (USA) where she worked on both hardware and software design for large business communication systems. She left research and development in electrical engineering to complete a Ph. D. in Greek in 1985.

3 Experiences with the IBM PC Network, and JANET/EARN access

W. B. DOWSLAND

*Department of Management Science and Statistics
University College of Swansea
Swansea, SA2 8PP*

Introduction

In September 1985 we took delivery of 13 IBM enhanced PC AT machines, together with IBM PC Network hardware and software. This equipment was set up as a laboratory to support a variety of undergraduate and postgraduate applications for our Operational Research, Management and Business Studies students. This included the support of programming languages (Fortran, Pascal, Basic) and for running a wide range of business software. In addition, it was necessary to provide terminal emulation facilities so that use could be made of a wide range of computer-based services both within and outside of Swansea.

The nature of the programming work meant that it was necessary to purchase a fairly powerful machine (hence the enhanced IBM PC AT machines), and the variety of resources (printers, plotter, laser printer, etc.) which were to be connected, together with the file server requirements, necessitated the interconnection using a local area network.

This paper describes our experiences with the network and outlines some of the ways in which the equipment has been used.

Network hardware

The IBM PC Network hardware consists of a network adapter board installed in each networked machine (IBM PC XT, or PC

AT), a translator/splitter unit (one for the entire network), and, where more than eight machines are involved, a base expander unit together with distance kits. It can provide support for up to 72 machines (nodes) using a star topology, and nodes can be positioned up to 1000 feet from the translator unit. Coaxial cable is used to connect the components together.

Network software

Two software products were available through IBM to support the network hardware, these being the IBM PC LAN program (an improved version of the earlier IBM PC Network program) and the Tapestry networking software. These two systems utilize exactly the same hardware but provide significantly different systems from a network manager and user points of view.

The IBM PC LAN program is essentially a low-cost software product installed on each machine on the network in one of four configurations: Server, Messenger, Receiver, or Redirector. The Server configuration allows sharing of disks, directories, and printers of the server machine whereas the three other configurations all allow access to any shared resources. These three differ from each other in respect of the message facilities available and the user interface provided to the Network program. In the network each node is a peer of the others. The only network management function available is from server machines, whose device availability and usage can be monitored. Several printers (both LPT and COM) and directories can be shared, password protection can be provided, and access to disks and directories can be restricted to read only or read write, etc.

The second networking system, the Tapestry networking software is primarily an icon-driven system which is organized around a network manager station which authorizes user connections to the system and provides electronic mail facilities. Several forms of server machine can be defined, including an asynchronous communications server for modem access, and requests for communications devices are automatically routed to appropriate free devices. The mail facilities are far more sophisticated than the message facilities of IBM PC LAN. However, one restriction imposed by Tapestry is that a print server may only share one device at a time with the network.

In both cases communications around the network are on a

parallel basis (LPT1, 2, and 3), though serial devices can be attached and shared by server machines using appropriate MODE commands. Server and manager machines can be used as workstations and are not dedicated, though the performance of these machines is obviously affected by network load. In the case of the IBM Network program the priorities given to tasks on a server can be adjusted to meet local needs.

Our selection of the IBM network product was based partly on cost, and partly because the communications server facilities offered by the Tapestry networking system were not a critical element in our own configuration.

Setting up a network

Prior to delivery of the software and hardware, suitable cable trunking had been installed for the network coaxial cable, and cabling had been installed to provide links from machine serial ports to a local PAD, and thereby providing external communications capability.

The network configuration chosen is shown in Fig. 3.1. Two non-dedicated server machines are in use, one acting as a file server and

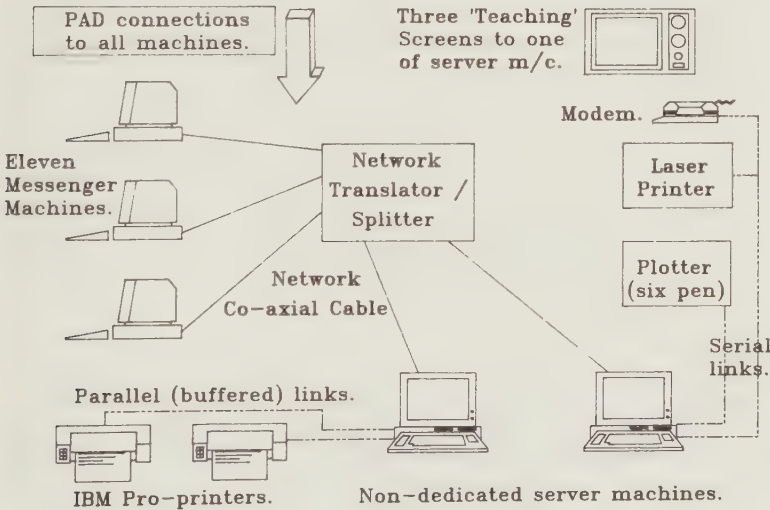


Fig. 3.1 IBM Network configuration.

supporting two shared IBM dot matrix Pro-printers. The second server provides plotter and laser printing facilities. Eleven other machines are configured to use these servers, though it would only require simple software changes to enable any network machine to act as a server.

The IBM LAN program comes complete with a software installation aid which is run following the installation of the hardware. This creates an appropriate file structure, copies DOS 3.1 and Network programs into suitable directories, and enables the installation of IBM applications software. It provides a very quick and easy means of setting up both server and messenger configurations.

The directory structure proved to be quite acceptable in our environment and we chose to install the more frequently accessed application software on *All* the machines. This included Fortran, Basic, word processing, spreadsheet, and database systems, as well as terminal emulation software. Other software, which we felt would be used less frequently, was installed on the file server. The network is set up so that a portion of the file server hard disk is always available to users (on a read only basis) as drive D.

In order to provide some degree of protection from 'innocent' system damage, certain DOS routines (FORMAT, FDISK, ATTRIB, and DEBUG) were removed from all machines except for a password protected portion of the file server disk. This was not intended to provide protection from malicious users, as this can never be fully provided with such a system. All users were 'forced' to make use of the high capacity floppy disk drive (A:) for their personal file storage, this being done partly using default data drive specifications within application programs, and partly through a policy of erasure of any 'foreign' files found on the hard disk of any machine. All application programs were protected using the DOS ATTRIB command making the files read only.

Points to consider

The system has been in use a year at the time of writing and has proved to be very reliable and 'student proof'. No major problems have been experienced, though there are a few points which should be considered when selecting the hardware configuration and before making software purchases.

Printer server machines

The decision to use a single machine to provide all primary printer services was taken so that the printer server machine could be left permanently up and running, thereby eliminating any user confusion over starting up and use of the network. This approach has proved very successful, except for some initial problems with the print queue. All devices shared by a single server make use of the same *single* output spooler. Thus, if the printing of a large file is in progress via a particular server, other devices being shared by that server are 'held' until the printing file has been completely downloaded to the printer. The IBM Pro-printers' text buffer meant that attempts to alleviate the 'single queue' problem using secondary software print spoolers proved unsuccessful. Finally, an in-line hardware buffer was introduced on each printer which has nearly eliminated printer queue problems and has, at the same time, improved the response experienced by a user of the server machine during printing.

Software considerations

A factor relating to software operation which needs to be considered is that all signals around the network are parallel (LPT1, 2, or 3). Some software cannot be configured to send output to parallel devices, this being particularly the case with software which provides plotter output, where COM1 or COM2 output is more usual. Plotters and other serial devices can be shared on the network but it is necessary to ensure that information to be sent to them is routed around the network using parallel communications channels.

Another important factor is that both network software products require a significant amount of memory for operation. In the case of the IBM PC LAN program installed on a 640 kbyte machine, this reduces the maximum memory available for programs to around 400 kbyte with a typical server configuration. If a redirector configuration (the simplest) is used this rises to around 560 kbyte. In our set-up the majority of machines have been installed using a Messenger configuration (leaving around 440 kbyte), this providing a menu-driven interface to perform network commands. In view of the increasing demands placed by software on memory, the reductions outlined above could easily prove restrictive for certain

software products and this is obviously an area for future development.

The IBM PC LAN program does provide the capability of both file and record locking and this can be utilized by multi-user software (e.g. dBASE III +) and by some language compilers. However, a great deal of software does not currently support these locking facilities, and, in general, our approach to the use of the file server has been to transfer information and programs to the local machine for storage and processing, rather than to use it in a true multi-user mode. This will clearly change as reliable multi-user software becomes available.

Outside connections

Although for much of the time the network machines were intended to operate in a stand-alone mode, it was considered essential to provide links to other systems. In the university environment this can often be done most economically via JANET (the British **J**oint **A**cademic **N**ETwork) and PSS/IPSS which between them provide links to a wide range of computing and data services. The links through the serial port of each machine to a PAD and thereby to the campus network and JANET, provided this connection.

In order to make effective use of these links, software was needed to provide VT100 and Tektronix emulation, and for support of viewdata services such as Prestel. We make use of the emulation product Smarterm which provides both VT100 and Tektronix 4014 support. This, and similar products, provide emulation and file transfer as well as network plotting and printing facilities. For viewdata format support the product ChitChat is used, either using a modem or via PSS, depending upon the service being accessed. There are, of course, a number of very good public domain emulation products which provide many of the features of the above commercial products. Included amongst these are PC-VT, Procomm, and Tekterm2. User groups such as the IBM PC User Group (PO Box 830, London, SE1 2BQ) can provide information on the wide range of public domain software.

Services available

'Free' services

Connection from a networked IBM PC AT machine to external

services will usually take place via JANET, though we do make some use of an autodial modem for services which cannot presently be accessed in this way. JANET was inaugurated in April 1984 and now provides links to all university sites, as well as research establishments and a number of polytechnics on SERC research contracts. The access provided not only allows suitably authorized staff to make use of computer resources at other universities but also enables reliable and quick file transfer between machines. The facilities available are not restricted to computer processing, and recently a number of other services have become available. These include access to library holdings, bookshop stocks, and bulletin boards, as well as access to PSS and IPSS and other academic networks such as EARN (the **E**uropean **A**cademic **R**esearch **N**ETwork) and in North America BITNET and NETNORTH.

The EARN system is an international computer network between academic institutions which provides world-wide communications facilities. It consists of independent host nodes, with a central computer in each country providing some central services. It is currently financed by IBM, though they play no part in the management of the operation, and it is connected to several other networks, including BITNET in the USA, and NETNORTH in Canada. It is linked to the JANET network via a gateway at RAL (the **R**utherford **A**ppleton **L**aboratory). There are currently over 350 nodes attached to EARN outside the UK, and nearly 900 others on linked networks, in particular in the United States. Access to sites in Australasia, Japan, and the Middle East is also available, and there is a free message service between users on any of the nodes. However, a problem still remains in determining the universities and individuals who can be contacted in this way.

Chargeable services

The more commercial services, which are not available on JANET itself, can frequently be accessed via the PSS/IPSS system. Two gateways from JANET (at RAL and University of London Computing Centre ULCC) provide this access. Charges for PSS/IPSS are based upon both connect time and on the volumes of data sent and received (these costs are generally low), and the user is also subject to commercial charges which are generally incurred from the host services.

The IPSS link enables access to a wide range of sites throughout the world, and a number of these are illustrated to our Business and

Management students. The ECHO organization (**E**uropean **C**ommission **H**ost **O**rganisation) provides details of many of the European services available. These include:

- (1) **DIANE (D**irect **I**nformation **A**ccess **N**etwork for **E**urope), which brings together the major database hosts and telecommunication authorities to provide online access to over 400 different databases on over 50 host systems;
- (2) **EURODICAUTOM**, which provides European language support for technical terms in common use in the community;
- (3) **EUREKA DATA**, which provides information on academics and researchers together with projects concerned with European integration;
- (4) **TED (T**enders **E**lectronic **D**aily), which is an online list of all invitations to tender for European Community projects.

In addition to the above there is, of course, a vast array of fully commercial services available. Amongst those which we find useful are:

1. **DIALOG**, which is a USA-based system hosting over 200 separate databases held on a number of linked mainframe computers. Like many US-based systems, it is somewhat biased in the range of information provided but does include amongst its databases British Company Directory and Information Science Abstracts. Access can be expensive with some databases costing up to £100 an hour to access.
2. **Pergamon Infoline**, an international online service operating both in Britain and the USA but available in many other countries. It is biased rather more towards UK needs, and includes amongst over 60 databases, a number of interest to business users. These include electronic publishing abstracts, Dun and Bradstreet's Key British Enterprises, management and marketing abstracts, and the Jordanwatch Company Information Service.

In addition to these services, the more familiar facilities of Prestel and Telecom Gold are available (either via PSS or using a modem), as are the many bulletin board services.

Conclusions

The IBM PC Network hardware and software has proved to be easy both to install and operate. It works very much at a DOS level, rather than making use of icons, and has proved to be a very effective low-cost system. It is, however, restrictive in respect of its messaging facilities and also does not provide an asynchronous communications server.

The shared resources provided on the network have enabled a wide range of applications to be provided for users, and for rapid software upgrade to be undertaken. Future developments, including the IBM Token Ring Network and communications advances with asynchronous and X25 support, will no doubt lead to local area networks becoming a natural part of the IBM PC environment.



BILL DOWSLAND

Bill Dowsland is a lecturer in the Management Science and Statistics Department at the University College of Swansea. He obtained a first class honours degree in industrial engineering in 1973 and his Ph. D. in Management Information Systems in 1977. His interests include the application of computer technology to managerial and operational decision making and his recent work has been principally concerned with the application of operational research techniques in product design and distribution. He has published principally in the operational research field and has provided consultancy services in operational research and computing to a wide range of companies.

4 A dialogue processor on the IBM PC

J. WHALLEY

*Department of Geology
Portsmouth Polytechnic
Portsmouth, PO1 3QL*

D. RUSH

*Department of Mathematics, Statistics and Computing
Liverpool Polytechnic
Byrom Street
Liverpool, L3 3AF*

Introduction

Most computer programs involve the user in some form of interaction with the system. In computing's early days such 'dialogues' were strongly constrained by the input/output devices through which they were conducted, and in practice they were as convenient and intelligible as Morse code. Hardware restrictions have long since been eased, but the languages in which we write our programs frequently betray their age through their I/O (Input/Output) facilities. The full range of I/O capabilities offered by modern microcomputers is often only directly accessible by resorting to assembler programming, or by using obscure extensions to a high level language. This makes the task of I/O programming so time consuming that, today, when a wide range of facilities exists which can help to create a friendly and informative user interface, our programs are often, at best, obscure in their relations with the user and, at worst, positively user-hostile!

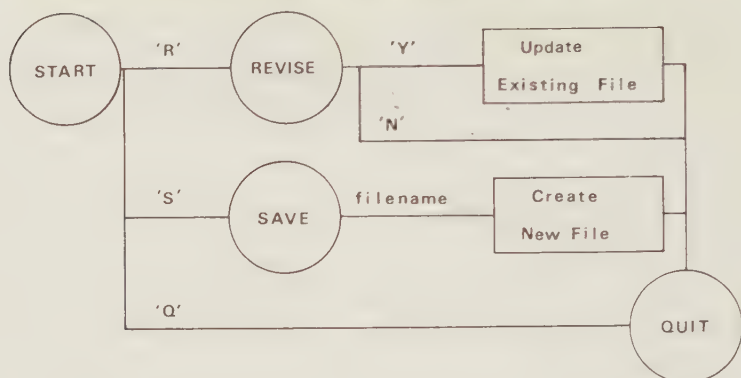
This is ironic since many computer scientists now believe that the user interface, far from being a subsidiary portion of a computer system, should be the primary target for the design and development process. The implementation of this philosophy has resulted in the development of software known as dialogue processors.

In this article we will describe the nature of dialogue processors in general and of MicroSYNICS in particular. This is a simple processor written in Basic for the IBM PC. We will briefly examine three applications of MicroSYNICS which illustrate the range of uses to which it is suited.

Dialogue processors

The importance of the user interface is emphasized by structuring the program into two interconnected components: the processing component, and the user interface component. This latter functions as a communications channel connecting the user and the processing component; all messages between them pass through it, and it may be constructed with a software tool known as a dialogue processor. When the dialogue processor approach is adopted, the starting point to program development is the specification of the user interface, i.e. the range of inputs from the user and the corresponding outputs from the processing component. The dialogue itself may then be constructed and tested. At this stage the processing component is simply a destination for, or source of, messages and does not need to be specified or implemented to any significant level of detail. Finally, when the face that the program shows to the user is judged to be correct, then, and only then, is it time to implement the processing component.

To specify a user interface, some particular notation must be adopted. Several workers have independently chosen the transition state diagram for this purpose. A typical, simple transition state diagram (Fig. 4.1) consists of a set of circles, the nodes, linked by arcs. Each node represents a program state which may be reached during a dialogue between user and program. The course of a dialogue corresponds to movement across the diagram; each interaction moves the program from a state represented by some node to a state represented by a linked node. Upon entry to a node, the program displays some specific text and then records the user's response. Attached to each arc is a short text string, and the user's response is compared to all strings for the current node. A transition then takes place along the arc for which the pre-stored text matches the user response. This simple scheme may be elaborated in a variety of ways. Greater flexibility may be introduced into the specification both of text displayed by a node and of the processing of user responses. For example, facilities may be provided to present text in



```

node START
,   File Update
,   Please choose one of
,   R — Revise a File
,   S — Save a File
,   Q — Quit
,   Command?

```

```

node REVISE
,   Are You Sure?

```

```

node SAVE
,   Give Filename

```

Fig. 4.1 A simple transition state diagram.

a variety of screen layouts, with partitioned screens or windows. In analysing input, it should be possible to accept text consisting not only of printable characters, but also the whole range of control characters and function keys. A very useful facility is the ability to accept single characters rather than inputs terminated by a carriage return.

Perhaps the most useful extension can be diagram decomposition. This refers to the provision of a symbol, the presence of which indicates a complete subconversation, perhaps indicating a whole diagram in itself. The inclusion of such a symbol adds a hierarchical structure to a diagram, the subconversation being called from the main diagram and returning values to it. There are obvious advantages to such an approach in that it makes diagrams more comprehensible, modifiable, and maintainable.

MicroSYNICS

MicroSYNICS text directly implements the simplest of the

notations described above and consists of a network of nodes. Within a node, input from the user can be accepted in response to prompts, and then transfer between nodes can be controlled by testing for the presence or absence of particular strings in the input. The transfer can be a simple branch or can use a `CALL/RETURN` syntax. A powerful feature of MicroSYNICS is its ability to execute a Basic subprogram from within a dialogue, and these subprograms can pass messages back to MicroSYNICS which can then be tested in exactly the same way as user input. Because the MicroSYNICS run time system is itself a Basic program, its internal variables can be manipulated by user-supplied subprograms, thus allowing the programmer to extend the existing set of condition testing facilities. Fig. 4.2 shows the dialogue of Fig. 4.1 encoded in MicroSYNICS.

Example applications

A mineral identification system

The syntax of MicroSYNICS is sufficiently extensive to allow a useful dialogue to be created without resorting to Basic subprograms. Starting from a standard flowchart used by Open University students for mineral identification, we have implemented a dialogue which guides the user towards distinguishing between 10 common minerals. MicroSYNICS is well suited to this type of application.

The dialogue poses a series of questions about the physical properties of minerals, the next question being determined on the basis of previous responses. In this way the system can exhibit a certain measure of intelligence in the selection of questions. However, it must be emphasized that dialogue processors do not provide the facilities expected of even the most primitive expert system shells.

A tutorial about programming

A good way for students to improve their programming is by analysing the text and behaviour of other people's programs. To gain benefit from doing this they must try to answer specific questions about program construction and behaviour, a difficult process for a lecturer to supervise. However, a program to be investigated may easily be embedded as a subprogram within a


```

REMARK.....
NODE START
; File Update
; Please choose one of:
; R—Revise a file
; S—Save a file
; Q—Quit

TO REVISE IF
@ 'R'
TO SAVE IF
@ 'S'
TO QUIT IF
@ 'Q'

REMARK.....
NODE REVISE
; Are you sure (Y/N)?
TO QUIT IF
@ 'Y' ['UPDATE']/'N'

REMARK
NODE SAVE
; Give filename
EXECUTE 'GETNAME'
EXECUTE 'FILESAVE'
GOTO QUIT

REMARK.....
NODE QUIT
; End of example
EXIT
*
```

Fig. 4.2 A MicroSYNICS translation of Fig. 4.1. All text on lines commencing with ';' is output to the screen. The '@' marks the start of a condition; the condition being that the following string appears in the current input line. The strings 'UPDATE', 'GETNAME', and 'FILESAVE' are the names of Basic subprograms. The REMARK command is analogous to the REM statement in Basic.

MicroSYNICS dialogue, which can then act as a guide through a sequence of tests.

We have applied this approach to a simple, published Basic program which calculates VAT. The chosen program is seriously flawed in that it does not validate its input and encounters rounding error problems. Specifically, negative rates of VAT and item costs are accepted, and the sum of the item cost and VAT payable can differ by 1p from the computed total cost.

A MicroSYNICS dialogue, which is a tutorial on the program's deficiencies, has been constructed. After a brief introduction the student is invited to run the program using a variety of data, to gain an appreciation of its intended operation. Then the student is directed to run the program with data which illustrates its lack of input validation and so is led to an exercise on writing appropriate validation statements. Finally, the program is run directly by the dialogue, with data which exemplifies the rounding error problem.

The MicroSYNICS dialogue for this application is constructed as a linear sequence of nodes. Each stage is encoded within one or more nodes. The only use for conditional branching is to allow a student to retry a stage. It had been hoped to use the VAT program unmodified, but this was not possible. For each of the separate interaction stages, e.g. change VAT?, input another item?, the tutorial dialogue must activate a separate program.

Improving an existing program interface

We were recently asked to translate a program which performed geochemical calculations and which had originally been written to operate in batch mode on non-IBM minicomputers. The program's input was by means of DATA statements embedded in the text and these naturally required editing each time the program was run. Few changes were required to the Basic in order to run on an IBM PC but considerable improvements to the user interface were felt to be necessary.

Instead of re-writing the Basic code it was decided to embed it within a MicroSYNICS dialogue, omitting only the READ and DATA statements. Additionally it was necessary to renumber the lines of the original program to start at 10 000. The dialogue allows all data input to be in response to explicit and informative prompts. Additionally, the data can be validated and, if necessary, remedial action taken. During execution, the MicroSYNICS runtime system and the embedded program are effectively two parts of the same

program, and so share all variables. Consequently, at the time that the embedded program is invoked, its variables have already been given their required values during the dialogue phase and no further input is needed.

At the end of processing by the embedded program, control is passed back to MicroSYNICS, and users are prompted for new or amended data before running the program again.

The future of dialogue processors

It must be said that the arguments in favour of dialogue processors are not universally accepted. Questions are often raised as to whether it is always possible to separate completely the user interface from application processing. In spite of these doubts there is much active research into the development of dialogue processors. Some workers have extended the two-component model to three components and split what we have described as the dialogue processor into an I/O presenter to generate images and read in raw input, and a separate, device independent, dialogue controller. In this model, the operation of the three components is then controlled by an overall User Interface Manager (UIM). Another development at the theoretical level is the use of alternative formalisms such as Augmented Transition Networks (ATNs), rather than transition diagrams.

Much attention is being paid to improvements to the instrumentation of dialogue processors, and facilities to view and manipulate state diagrams during development are being provided. User activity monitoring, and dialogue tracing, are also now seen as necessary. One system offers the ability to trace diagrams in one window while interaction takes place with the application in another.

Conclusions

There are now several dialogue processors, all based upon a transition state diagram notation. None, as yet, is widely available commercially and MicroSYNICS is by no means the most comprehensive in the facilities it offers. Perhaps its major deficiency is that it provides an interface to only one programming language, Basic. Another is that it is implemented in such a way that any subprogram can communicate directly with the user, thus bypassing

the input/output control of the dialogue processor. However, it is a useful vehicle for illustrating the principles of dialogue processors, and as our examples have shown, it is in fact sufficiently comprehensive to provide a range of useful facilities.



DAVID RUSH

Although he took a first degree in chemistry, after graduation David Rush rapidly moved into computing via postgraduate research in computational chemistry. Since 1966 he has held appointments both as a lecturer and as a member of the computer services units at three polytechnics: City of London, Teeside, and Portsmouth. He has recently been appointed Head of the Department of Mathematics, Statistics, and Computing, at Liverpool Polytechnic. His main research interests are in the application of artificial intelligence techniques to the improvement of the user interface, and to the creation of automated teaching systems.



JOHN WHALLEY

John Whalley graduated in geology from Leicester University and, after a period with the National Coal Board, returned to the academic fold with postgraduate research at Imperial College and the University of Leeds. Since 1979 he has been Senior Lecturer in Geology at Portsmouth Polytechnic,

specializing in structural geology and computer applications in geology. His current research interests are in the tectonics of south-west England and in the application of database and graphical techniques to the processing of three-dimensional geological data.

5 Large-scale examination processing on an IBM PC AT

P. ALLATT

*Computing Services
University of Salford
Salford, M5 4WT*

J. B. SLATER

*SWURCC,
University of Bath
Bath BA2 2AY*

M. VASSILOGLOU

*Computing Services
University of Salford
Salford, M5 4WT*

Introduction

Public examinations play a central role in the British educational system. Each year, hundreds of thousands of candidates take examinations such as the GCE, CSE, and the new GCSE, with students entering for examinations on a subject basis. The examination boards analyse and combine vast amounts of data in the process of assigning grades and do their best to ensure that all examination work is marked fairly and consistently. They attempt to award final grades which are sufficiently detached from the particular teaching, syllabus, examination paper, etc., so as to have meaning to those receiving and using them.

The examining process involves a number of individuals, such as markers, senior examiners, and permanent staff of the boards. When teams of markers are used in an examination, senior examiners need to satisfy themselves that the markers interpret the marking schemes consistently, so various procedures are employed, such as double marking of scripts and, most importantly, co-ordination meetings, in order to produce a high level of consensus. If necessary, the marks of some of the markers are adjusted appropriately.

Once this stage is complete, senior examiners use statistical information in an attempt to reinforce their impressions of how the examination has performed. Marks are aggregated, usually by simple or weighted addition, and candidates are often placed in an overall rank order. Finally, the examiners assign grades to the candidates: first they define requirements for the grades, usually expressed in terms of aggregate marks (although minimum requirements, or hurdles, on parts of the examination can also be included); then, candidates needing special attention are scrutinized in greater detail. Examples include those candidates close to key grade boundaries, those with unusual patterns of achievement, and those for whom a second opinion has been sought by the marker.

The Secondary Examinations Council is funding a project called Decision Analytic Aids to Examining, whose objective is to explore ways in which the examiners can be better supported in their tasks. The project is producing an interactive software package on IBM workstations, incorporating techniques from exploratory data analysis and decision analysis.

The 'Decision Analytic Aids to Examining' project

Exploratory data analysis has its foundation in the work of Tukey (1977). Several of the pictorial presentations described in this work are included in the package to enable important features of the data to be appreciated by examiners who are not statistically trained. Decision analysis was initially used in business applications to facilitate management planning; its use in the examination context is proposed in the work of French (1981) and Vassiloglou (1985). The two techniques combine to provide insight as to how an examination has performed and to guide examiners in forming, clarifying, and refining their judgements in a consistent fashion.

An examination is viewed as a tree structure of its components, including papers, sections, and questions. Candidate marks may be stored at any level on the tree and typically are collected at question level. The package provides facilities to define and select candidate groups in terms of various criteria, such as which marker has assessed them on a component, and which components they have attempted. Examiners can view and compare mark distributions for selected groups using different displays, for example scatterplots, histograms, boxplots, and numerical summaries. Other facilities

available in the package include mark adjustment, different forms of mark aggregation, assignment of grades, and flagging of candidates who require special attention.

Software

Although the project is essentially research orientated, software is produced to a professional standard. This includes the vital human computer interface, which is menu-driven with subsidiary dialogue. To provide portability, the project chose to write in standard Fortran 77 and use GKS (the Graphics Kernel System) for graphical output. Rapid access to data is achieved via a doubly linked list file structure based on Fortran direct access files, and a small number of interface routines are provided to position files and read or write records.

The IBM Professional Fortran compiler by Ryan-McFarland provides a complete implementation of the full 77 standard. Two widely available extensions are used: half-word integers to reduce storage requirements, and the INCLUDE feature to handle common block and parameter definitions. In general, it produces code that works efficiently on the chosen hardware.

The implementation of GKS used is that provided by IBM in the Professional Graphics series. Although not a full implementation of any formal level of the GKS standard, it contains features from several levels and it provides sufficient facilities to produce menus, interactive dialogue, and the various types of plot. The relatively low resolution of the IBM Enhanced Colour Display imposes restrictions on the number of plots which can be displayed simultaneously, making some comparisons difficult, but this might be solved with IBM's Professional Graphics Display.

Source code is maintained using the screen-based IBM Personal Editor; its ability to handle several files and to cut and paste, both within a given file and between files, having proved most useful. Object modules are stored and maintained using libraries, with the Linker giving the ability to replace library versions of specific routines with new versions, thus speeding development and debugging.

The choice of Fortran 77 and GKS has necessitated the use of the DOS operating system. This total package consists of some 200 user routines, together with routines from the Fortran and GKS libraries. The overheads of GKS and DOS, together with the data structures

used by the package, mean that the 640 K memory restriction of DOS is a problem. Using overlays could partially alleviate this, but it would compromise the portability of the package. Another possible solution is the use of Xenix, but currently GKS is not available in that environment.

Hardware

There are a number of factors influencing the choice of workstation. One is obviously the size of public examinations. A second is the need to provide information pictorially. Yet another is the need to provide rapid answers to such questions as 'How did the candidates who attempted Paper 2 Question 4, and were marked by X on Paper 1, do overall?' or 'Which markers appear out of line on Paper 1 Question 5?'. A large amount of disk space, significant processing power, and a graphics capability, are all needed to satisfy these requirements. Accordingly, the project chose to use an IBM PC AT with a maths coprocessor, the IBM Enhanced Graphics Adaptor, and an IBM Enhanced Colour Display. This machine has proved robust and easy to transport; one unit even survived a major car crash without loss of data!

In the future, with the introduction of the GCSE, there will be a need to support larger examinations. This in turn will demand more disk capacity, memory, and CPU power than is provided by an IBM PC AT under DOS. Thus, the project is considering implementing the package on the IBM 6151.

References

- French, S. (1981). Measurement theory and examinations. *British Journal of Mathematical and Statistical Psychology* **34**, 38–49.
- Tukey, J.W. (1977). *Exploratory data analysis*. Addison-Wesley, Reading, MA.
- Vassiloglou, M. (1985). Some multi-attribute models in examination assessment. *British Journal of Mathematical and Statistical Psychology* **37**, 216–33.

MARILENA VASSILOGLOU



Marilena Vassiloglou graduated in Decision Theory, from the University of Manchester in 1981. In her Master's and Ph.D. research she concentrated on the decision theoretic aspects of examination assessment. She has worked for two years on the SEC project 'Decision Analytic Aids to Examining'.

JOHN SLATER

John Slater, in his short and undistinguished career, read mathematics at Oxford, was a lecturer in pure mathematics at Oxford and London, moved into computing service provision, and is currently at Bath. He is involved with the University Grants Committee/Computer Board teaching initiative and had a short period of leave at Oxford, working on a variety of uses of computers in GCE and GCSE examinations, an area in which he has worked for longer than he cares to remember.

PAUL ALLATT

Paul Allatt graduated in mathematics from the University of London in 1975 and, after three years' research in number theory, he took up a post at Imperial College Computer Centre. For the past five years he has been head of the compiler group there, and is currently on secondment to the SEC project 'Decision Analytic Aids to Examining'.

6 Computer-assisted learning in chemistry, based on an IBM PC Network

The CAL Group*

*School of Chemistry
Liverpool University
PO BOX 147
Liverpool L69 3BX*

Background to the project

During the past two years the Computer Board for Universities and Research Councils, in conjunction with the University Grants Committee, has supported 106 pilot projects (the Computers in Teaching Initiative) developing the use of computers in undergraduate education over the full range of academic disciplines. Of the three universities which have received funding for work in chemistry, Liverpool has been granted by far the largest level of support.

The aims of the Liverpool project in chemistry are to introduce computer-assisted learning into the undergraduate chemistry courses in order to support, complement, and extend the practical and tutorial teaching. In particular, it is intended to teach the design and implementation of experiments including simulation, control, data capture and analysis; to introduce CAL programs in aspects of organic, physical, and structural inorganic chemistry; and to permit students to gain some experience in database manipulation and the use of Expert Systems. In order to ensure rapid evolution of the project from its inception in September 1985, and with the particular aim of having a working system available for undergraduate use by October 1986, just under half of the funds requested have been used

*The CAL GROUP consists of Drs D. J. Chadwick, D. L. Cooper, P. P. Duce, D. Margerison, S. M. Walker, and T. W. Walton.

to support two post-doctoral research assistants, the remaining monies being allocated to the purchase of hardware and software.

In the remainder of this article, the hardware configuration that has been adopted will be described first, followed by a report of progress in the development of software for the teaching of physical and organic chemistry.

Hardware configuration

The design of the undergraduate chemistry computing facility has been subject to two major considerations. The first has been to provide as many microcomputer workstations as possible. This would ensure that undergraduates would have ready and essentially open access to the system, thereby making this mode of instruction more acceptable to them. The extent to which this aim can be achieved depends upon the second consideration, which is inevitably cost: with only a limited budget for hardware and software, it is clear that the whole project design necessarily involves a fine balance between the two competing criteria. Difficulties are further exacerbated if secondary, albeit highly desirable, features are incorporated into the plan. For example, in practical physical chemistry the collection of experimental data over an extended period, and its subsequent analysis, is a common requirement: this suggested that some micros with the additional capability of online data logging should be sited in the teaching laboratories. In addition, professional chemists make extensive use of online databases for the storage and retrieval of structural and analytical information. These may be accessed through JANET (the British Joint Academic **NET**work). If undergraduates are to gain any experience in this area, then links have to be provided between the chemistry teaching facility and the University IBM 3083 mainframe. These provide the additional bonus of allowing all local disk backup requirements to be met by dumping to the mainframe disk storage, and thence to the mainframe archiving facilities, thus eliminating some of the routine housekeeping chores and saving valuable local storage.

Bearing all these considerations in mind, it is probably not surprising that the IBM PC Network environment was chosen, allowing the number of micros to be maximized by the sharing of expensive peripherals such as hard disks, printers and plotters, etc. Coincidentally (and most fortunately), at the time of planning IBM were able to offer very substantial discounts on the IBM Portable PC coupled with a cheap colour monitor (essential for chemistry

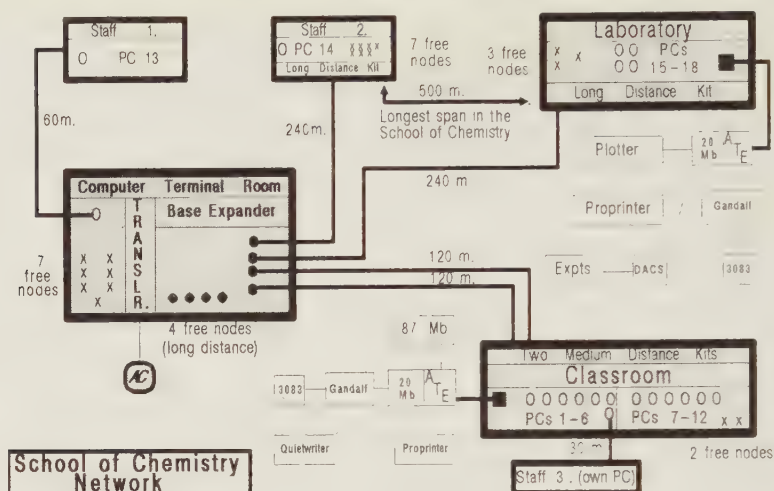


Fig. 6.1 School of Chemistry Network.

teaching with graphics) which could not be matched for performance or price by any other supplier. The choice of such a machine may, at first glance, appear somewhat surprising, but price considerations coupled with the recognition that the machine is essentially an IBM PC XT were compelling. In fact, the machines have proved to be perfectly hardy workhorses, acting as networked slaves to our two AT file servers. In the event, the deal offered by IBM allowed all the IBM PCs to be equipped with the appropriate maths coprocessor, thus vastly extending the potential for tackling the more complex problems in theoretical chemistry, in conjunction with the Professional Fortran package supplied by IBM.

Figure 6.1 shows the outline of the network as implemented in the School of Chemistry. The maximum physical distance allowed between extreme points of the net is 2 000 feet, which just covers the combined activities within the school; by placement of the translator unit centrally, it has proved possible to satisfy this limitation. Fortunately the room occupying this central position happens to house the computing terminals provided for research use with the mainframe, and there is ample room for expansion via the unused nodes connected directly to the translator. Two clusters of terminals have been established to satisfy the teaching requirements. The major classroom contains 12 IBM Portable PCs plus one IBM PC AT, while four IBM Portable PCs plus a further IBM PC AT have

been placed in the Physical Chemistry teaching laboratory. In addition, three members of staff are linked into the system through terminals in their offices, to facilitate software development and network management. It should be clear from Fig. 6.1 that the arrangement offers substantial room for further expansion by the creation of more classroom clusters and by linking of more offices, should funds permit. Both of the undergraduate facilities have an IBM Pro-printer for student use and draft document preparation; additionally, the classroom cluster has an IBM Quietwriter for the production of manuals, and an IBM 7371 plotter has been sited in the laboratory. The laboratory IBM PC AT has a data acquisition and control adapter card, plus support software for online data logging, and both IBM PC ATs have asynchronous connections to the IBM 3083 mainframe.

A review of the experience gained in designing, commissioning, and running, the network may perhaps be of help and interest to users thinking of following a similar path.

1. It takes very much longer than might be expected simply to transport, unpack, configure, and site 20 IBM PCs. The initial delivery weighed some 2.5 tons which had to be moved to the second floor of the Chemistry Building over a distance of about 200 yards. Each micro had to be disassembled for the installation of network cards, coprocessors and the like, and the machines then sited in locations so that students could reach them easily.
2. Cabling (for both power and the network) has to be safe, unobtrusive, and inaccessible to unauthorized personnel. The University authorities quite correctly required that this had to be carried out professionally which was an unexpected expense and took longer than planned.
3. It is essential to be over-generous in planning the size of the hard-disk storage. In the original grant submission, it had confidently been predicted that the two 20 Mbyte disks would satisfy the bulk storage needs for some considerable time. How wrong that has proved to be! Installation of all the applications on the disks consumed 4 Mbytes in a couple of days and now that software production has gained momentum, about 1 Mbyte is being consumed every two weeks! Hard-disk storage has recently been increased to include a further 87 Mbytes, sadly lack of funds prevents further expansion.
4. Perhaps the most important observation concerns access time. The speed of operation of the network is suitably fast and, in an

undergraduate environment, is not a limiting factor, but when 20 people are trying to access the same file on a hard disk then delays can become unacceptably long. One solution (which for obvious reasons is unattractive) would be to have multiple copies of frequently accessed material residing on separate disks. The following factors have been found to be important in reducing such delays (in rank order):

- (i) Do not allow files to become fragmented. The development of software necessarily takes place in a piecemeal fashion, and after final debugging it is essential to recreate the resultant files as a contiguous whole.
- (ii) Use the 'slave' IBM PCs in the configuration that allows the maximum free memory consistent with the network philosophy. Message facilities in the undergraduate environment are liable to abuse, so the majority of IBM PCs are configured to work in Redirector Mode, which allows the use of bigger (and more) buffers with a vast improvement in network performance.
- (iii) For classroom use it is appropriate to optimize the appropriate IBM PC AT server for background operation, effectively turning it into a dedicated server. Important foreground tasks can always be performed outside normal teaching hours.

Development of CAL software for chemistry

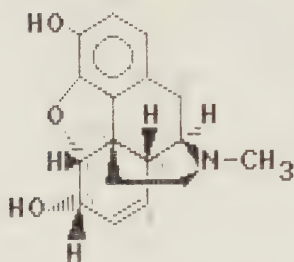
Organic chemistry

All the tutorial material for organic chemistry has thus far been prepared using the Domino authoring system produced by Compssoft PLC. This system provides text and graphics screen facilities, and accepts multiple choice, text, numeric, or date responses from the user. Numerical calculation is not possible within the system. For a given screen display, different responses simply direct the user to other screens. Up to 10 specific choices are allowed, plus a default for invalid replies. There is also the possibility of automatic scrolling through a sequence of screens. Given these features it has proved possible to design tutorials with branching tree-like or network structures. The system seems to lend itself quite well to the needs of descriptive organic chemistry.

The tutorial programs, when complete, are stored on one of the IBM PC AT hard disks and accessed by users from the 'slave' IBM PCs over the network. To date, two main kinds of tutorial, differing

Sample answer:

1R,2S,5S,etc.



Type in the complete assignment

[5R,6S,9R,13S,14R] []

Fig. 6.3 Screen dump of a 'letter response' to a question.

in their appearance to the user, have been developed.

First, there is the relatively straightforward 'linear' type of package, where the students are given some information, followed by a sequence of problems to test their understanding. If a wrong answer is entered, this is flagged (the background colour of the page changing from the customary restful blue to a vivid red) and supplementary problems are accessed. Depending on the students' performance, they may be returned to some earlier points in the tutorial for revision or, very occasionally and if their performance is consistently wrong, they may be routed out of the tutorial completely and told to go and read a book! The problems either expect a textual or numerical response, or require the user to choose from a range of options by placing the cursor in a box containing the appropriate figure. This latter type of response encourages interaction with the screen, giving more involvement with the tutorial. An example of this type of package is one designed to teach some aspects of stereochemical nomenclature. This is illustrated in Fig. 6.2-6.4; Fig. 6.2 showing the tree structure of part of the tutorial, Fig. 6.3 being a screen dump of a letter response to a question, and Fig. 6.4 a screen dump of a cursor in the box question. The tutorial, which is in three sections, takes about 75 minutes to work through in one sitting, while each section can, of course, be run individually.

Place the cursor in the box which contains the correct structure for (S)-2-pentanol.....then (only one attempt, so be sure!)

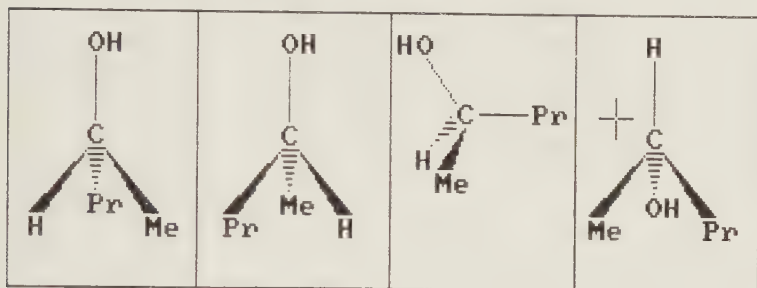


Fig. 6.4 Screen dump of a 'cursor in the box' question.

This type of tutorial structure is inevitably reminiscent of a textbook. This similarity is difficult to avoid in situations where new teaching information has to be supplied in quantity and where it is felt necessary to steer the user down a well-defined path to ensure that all the subject matter has been adequately rehearsed. It may be more appropriate in some circumstances to have a few screens of information available, perhaps as a 'help' option, in an otherwise problem-based tutorial. This raises a general question of how much factual knowledge the CAL package should assume the student has, and how much the package should be used to supply new information. The balance will depend on how CAL is viewed in relation to 'conventional' tutorials, lectures, and textbooks.

The second type of tutorial is a consequence of attempts to move away from the bookish appearance of the package, though the underlying structure is very similar to that described above. As part of a large package designed to teach the chemistry of aromatic substitution, some instruction along the lines already outlined is followed by a separate section consisting solely of synthesis problems. In these, the student is required to make a product from a given starting compound, using a set of 15 supplied reagents. As the correct choices are selected, the synthesis is developed across the screen. All 15 possible responses are coded for at each step. In response to an inappropriate choice of reagent, the background

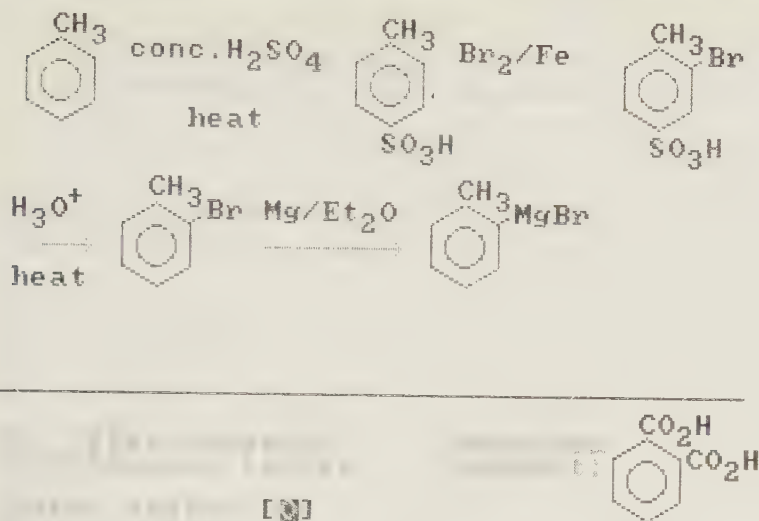


Fig. 6.6 Screen dump taken towards the end of a synthesis problem.

colour changes from blue to red and the students are told why the choice is unacceptable and what the probable chemical consequences of their choice would be. Some combinations of a fragile organic intermediate and a robust reagent (such as concentrated sulphuric acid) may produce a tar, and a black screen background is used to bring this point home! The tree structure behind such a tutorial is illustrated in Fig. 6.5, a screen dump taken towards the end of a synthesis is shown in Fig. 6.6, and the display of allowed reagents which is available on request throughout the tutorial is given in Fig. 6.7. In some instances, more than one acceptable route from starting material to product exists, and this has to be allowed for in the planning of appropriate responses. Similarly, students may start out along an acceptable path but then deviate into what ultimately becomes a dead end, in such a circumstance the program makes the users aware of the problem and eases them back into the correct path at the point where the deviation occurred. The synthesis problems have to be chosen and the programmed responses planned with great care lest the number of acceptable possible routes multiply alarmingly so that the problem becomes ultimately unmanageable. A series of problems with increasing numbers of steps has been constructed in this way, the final seven-step problem has just one

REAGENTS		
A $\text{H}_3\text{O}^+/\text{heat}$	F $\text{H}_3\text{PO}_2/\text{heat}$	K $\text{KI}/\text{H}_2\text{O}$
B $\text{HNO}_3/\text{H}_2\text{SO}_4$ heat	G $\text{NH}_4\text{HS}/\text{H}_2\text{O}$	L Br_2/Fe
C $\text{HNO}_3/\text{H}_2\text{SO}_4$ 5 °C	H HBF_4/heat	M $(\text{CH}_3\text{CO})_2\text{O}$
D conc. H_2SO_4 heat	I $\text{Sn}/\text{H}_3\text{O}^+$	N $\text{Mg}/\text{Et}_2\text{O}$
E $\text{HNO}_2/\text{H}_3\text{O}^+$ 5 °C	J $\text{KMnO}_4/\text{H}_3\text{O}^+$ heat	O CO_2 , then H_3O^+

Fig. 6.7 Display of allowed reagents available at request throughout the tutorial on synthesis problems.

correct route, all other potentially feasible routes being blocked at early stages to compel the student to home in on the correct strategy.

The possibilities afforded by the animation facilities within Domino have also been exploited. In a package on the chemistry of alkyl halides, some simple time-lapse sequences have been included which show the stereochemical consequences and associated energetics of two reaction mechanisms. This is an area of considerable potential since it is something which a book cannot illustrate.

The experience gained thus far has shown that, with some effort, adequate and attractive displays can be produced. All the structural drawings in the tutorial packages described above have been constructed from scratch by use of the box, ellipse, and line drawing facilities provided within the system. All subscripting has had to be done by pasting in individual characters while the 'curly arrows' representing electron movement, which are essential to the teaching of mechanistic organic chemistry, have had to be constructed from sections of ellipses. These limitations have meant that considerable time and effort has been expended in some cases to obtain accurate and aesthetically acceptable diagrams. However, the facility which

permits saving of images once constructed has led to the creation of an extensive library of re-usable blocks. The software also has an 'image gathering' feature whereby images produced by other packages can be saved and then manipulated within the editor. This has not been exploited as yet but may well prove a useful adjunct for the incorporation of more sophisticated molecular drawings generated by more powerful graphics software.

The tutorial packages consume large amounts of disk space. Each screen is stored as a file (for graphics screens the average size per screen is 5 Kbytes) and each set of screens is accompanied by two files which contain the reply, index, and tree structure information. A disadvantage is that each screen is allocated a set amount of disk space irrespective of whether or not it is all required for the information contained on the screen. Some additional run time files are also needed. The total disk space required e.g. for the stereo-chemistry package, comes to approximately 0.75 Mbytes. As originally constructed, the package on aromatic substitution occupied approximately 2 Mbytes, as a consequence of which disk search and access times became intolerably slow, particularly when several users were accessing the package simultaneously. This problem has to some extent been overcome by splitting the tutorial into smaller sections, a non-trivial task given the file structure created by the Domino system.

Physical chemistry

The physical chemistry component of the CAL group's activities initially involved the production of a package for the teaching of some of the basic ideas of quantum mechanics. First attempts utilizing Domino were quickly abandoned: the impossibility of incorporating facilities for numerical computation within a tutorial makes this authoring system totally unacceptable for this kind of task.

Since the chemistry CAL group at Leeds University under the direction of Professor P. B. Ayscough has produced software for the BBC Microcomputer based on P. W. Atkins's textbook of physical chemistry, it has been agreed that the Liverpool CAL group should undertake the task of converting this software to run on the IBM PC. Six packages of the 12 planned for the BBC Micro are close to completion, and one of these (Schrödinger Part 1) has now been converted for the IBM PC; a second, dealing with gas laws, is in the final stages of conversion, and work has begun on a third package covering elementary thermodynamics.

The general approach in these packages is to deal with simple systems illustrating the basic ideas, to represent the systems graphically, and to use animation wherever possible. The principle of one picture being worth a thousand words has been rigidly adhered to throughout. Thus, the quantum mechanics package, Schrödinger Part 1, simulates five simple one-dimensional systems illustrating a number of basic concepts, e.g. particle-wave duality, quantum mechanical tunnelling, and quantization of energy levels for bound particles. The five systems simulated are:

- (1) the effects of potential barriers on particles with a kinetic energy greater than the barrier's height;
- (2) the phenomenon of quantum mechanical tunnelling, i.e. the ability of particles to penetrate into and through classically forbidden regions;
- (3) electrons confined in a one-dimensional box with walls of finite height;
- (4) electrons confined in a one-dimensional box with walls of infinite height;
- (5) particles confined in a simple harmonic potential well.

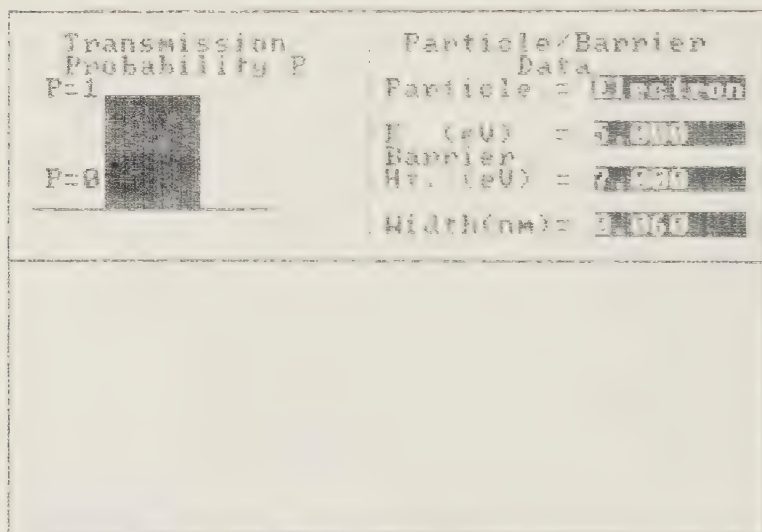
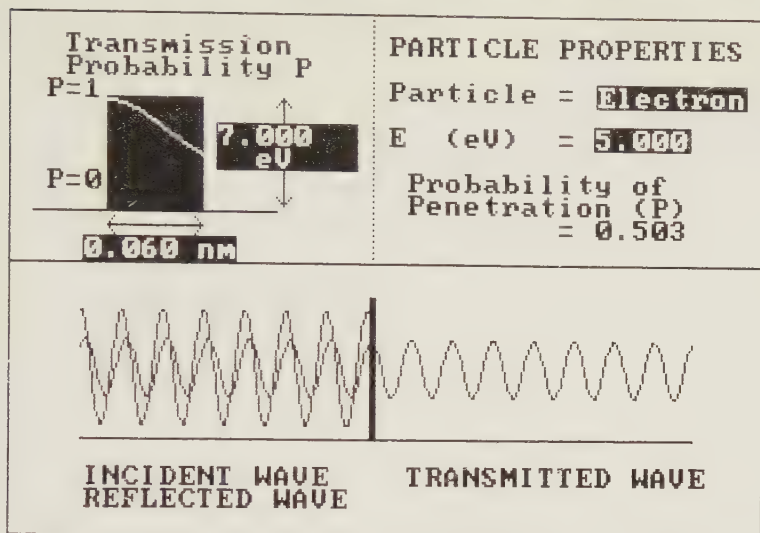


Fig. 6.8 Screen dump showing the parameters which may be varied by the student in the quantum mechanical tunnelling simulation.



<esc> to return to menu and R to restart

Fig. 6.9 Screen dump on running the simulation program with default values of particle and barrier parameters.

Utilizing these five systems, the package allows the student to explore the fundamental ideas outlined above. The chief advantage of this approach compared with that of a tutorial constructed with the Domino system is that students learn by doing rather than by being told. Thus, the student may vary the parameters for any of the five cases listed above, e.g. particle energy, system dimensions, etc. and determine how these influence the system. In the quantum mechanical tunnelling simulation, for example, the students may systematically vary barrier thickness and discover how this influences the probability of tunnelling; they can then repeat the exercise varying particle mass, or barrier height, or particle energy.

Figures 6.8–6.13 are screen dumps from three of the programs in the package: quantum mechanical tunnelling, simple harmonic potential well, and infinite potential well. Figure 6.8 shows the parameters which may be varied by the student in the tunnelling simulation, the top right hand corner of the display being in inverse video. Figure 6.9 is the display when the simulation program has been run with the default values of particle, particle energy, barrier height, and barrier thickness. Figure 6.10 shows the simple harmonic

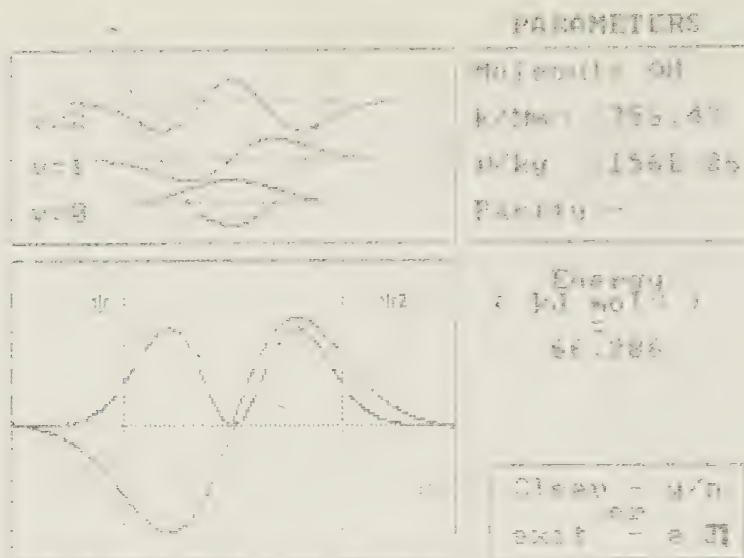


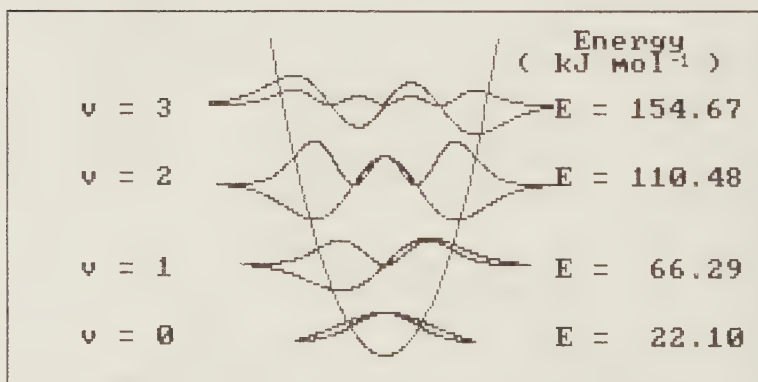
Fig. 6.10 Screen dump showing a simple harmonic potential well display (for details see text).

SIMPLE HARMONIC OSCILLATOR POTENTIAL

Molecule: OH

k/Nm^{-1} : 756

μ/kg : 0.1563E-26



<ESC> to return to menu R to restart

Fig. 6.11 End display for the simple harmonic potential well program.

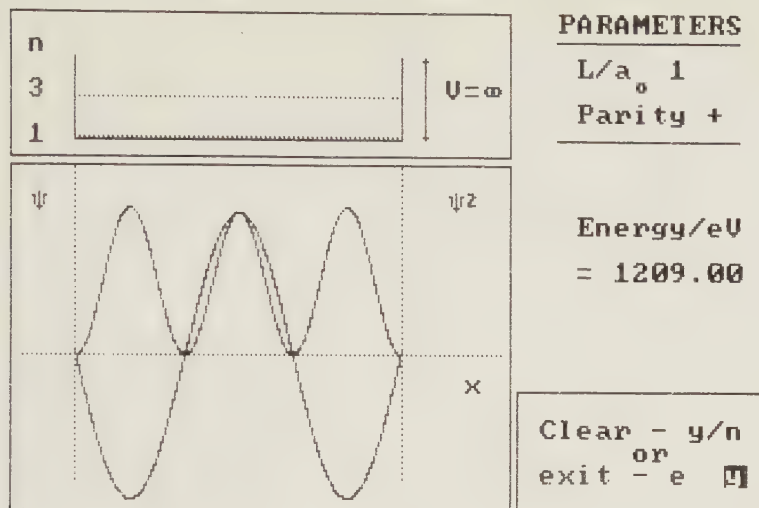
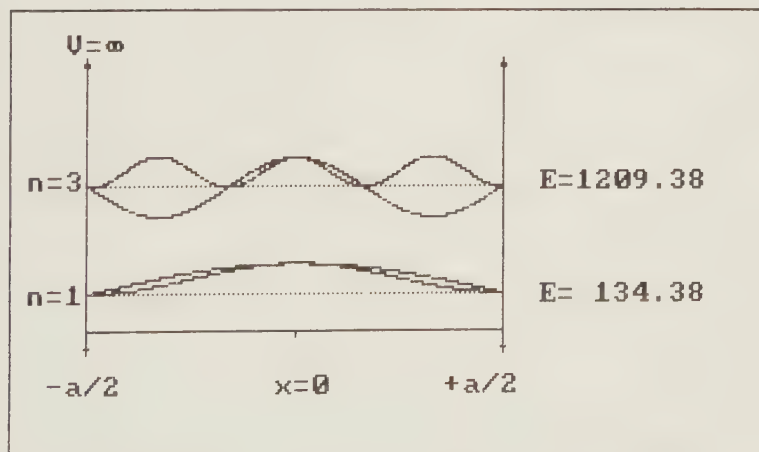


Fig. 6.12 Screen dump showing an infinite potential well display.

INFINITE SQUARE WELL POTENTIAL

$$\text{length}/a_0 = 1$$



<esc> to menu or R to restart

Fig. 6.13 End display for the infinite potential well program.

well display for the O-H bond indicating the second energy level. The box in the top left-hand corner of the screen dump displays which of the first four energy levels the user has found. The box below this contains a plot of the wave function and its square for the second energy level (in the actual display these are in different colours). Figure 6.11 is the end display for this program, and shows on a larger scale which energy levels amongst the first four have been found. Figures 6.12 and 6.13 are screen dumps of the analogous displays for the infinite potential well simulation, the plots in Fig. 6.12 of wave function and wave function squared being for the third energy level.

The original BBC version of this package was written in Basic. Of the two compilers available to this project at Liverpool: Fortran 77 with GKS and Microsoft Basic version 2, the latter has been selected for two reasons. First, the compiled programs are sufficiently fast for the tutorial requirements, indeed in some instances it has been necessary to slow programs down through incorporation of delay loops in order to enable the display to be followed. Secondly, if the package were to have been written in Fortran it would have been necessary to utilize GKS which involves large memory overheads; a typical Fortran program using GKS will occupy between three and four times as much memory as an equivalent Basic program.

Conclusions

Although this project has only been running for approximately one year (with the research assistants present for roughly half that time) great progress has been achieved. The hardware configuration is complete and ready for student use and a considerable amount of software has now been written. The urgent task, before the start of the next academic year and the general release of the system to undergraduates, is the construction of a menu-driven front end to the tutorials which will maintain general security of access via user names and passwords, log usage, prevent unauthorized interference with the system, and shield users from DOS. This is well in hand. Most of the available resources have been concentrated thus far on the preparation of a viable tutorial system; only recently has a start been made on the second aspect of this project, namely the teaching of experiment design and control.

For the future, it is possible to identify an enormous number of areas which lend themselves to the unique features afforded by CAL

techniques. The powerful colour graphics capabilities, the facilities for animation and real time calculation, and the ability of a program to respond in a user-sensitive manner give the CAL approach great potential as a pedagogical tool. The limiting factor quite simply is money: there is no shortage of ideas or enthusiasm, only of the resources to implement them. It is to be hoped that projects such as the one described here will be able to attract further financial support once the 'pump priming' phase is over, so that the expertise gained at considerable expense and effort is not lost.



The CAL GROUP, Liverpool University's School of Chemistry

This group is represented by Dr Chadwick, who should be contacted first if you wish to discuss the article or its contents, but the article is a contribution from the complete group which includes not only Dr Chadwick, a senior lecturer in organic chemistry, but also Drs Margerison, Walker, and Cooper from physical chemistry, and Drs Duce and Walton who are post-doctoral research assistants.

The Liverpool School of Chemistry comprises 28 academic staff and corresponding numbers of support staff, a range of instrumentation second to none in the UK, and a large and internationally famous research programme. Its current undergraduate intake is approximately 80 full-time students of chemistry, but it also provides courses for students of medicine and biochemistry and supports interdisciplinary courses in chemical physics, marine chemistry, chemistry and pharmacology, and chemistry with materials science.

The physical chemistry software described in this article is being published by Oxford University Press in the early part of 1988 as the Library of Physical Chemistry Software.

7 Developing expert systems for biological keys on the IBM PC

J. SCHOFIELD

*Bedford College of Higher Education
Cauldwell St
Bedford, MK42 9AH*

Introduction

The development of biological keys for identification purposes, particularly in areas where knowledge is incomplete, or in cases where the taxonomic framework is under review, pose many time consuming difficulties for the scientists involved. Any classification systems built up on the basis of dichotomous, or lateral, tree structures are often subject to radical modifications, where substructures are bodily moved from one place to another, new amplifications are inserted where none existed before, and detailed changes in textual descriptions are constantly demanded. These problems suggest the use of computer-based systems providing easy movement about the tree, word-processing features for editing text, and facilities for implementing major or minor changes in structure with the minimum difficulty. Indeed, several such systems have been developed on mainframe systems. The advent of inexpensive yet powerful micros, such as the IBM PC, have made the tackling of such problems possible on a much smaller budget. In addition, they have allowed such facilities as a wide colour range and powerful computer graphics to be brought to bear on such problems. The portability of an IBM PC has also meant that it can be much more accessible to the user, when on field trips.

The information available to users of such a key, who are looking for the best classification they can get, is frequently incomplete. This suggests the use of expert system type searches of the tree to

overcome any consequent ambiguities, by both traversing further sections of the tree and by delivering any identified sequences of alternative solutions. In addition, particular words used in the system might need explaining, requiring the provision of some kind of glossary.

Development of biological keys

It was considered necessary to provide separate programs to BUILD and ACCESS the trees. The first function is fulfilled by the provision of routines to CONSTRUCT, SAVE on disk, MODIFY, UPDATE, EXTEND, and PRINT dichotomous trees, while ACCESS to the tree would be provided by a separate TREE TRAVERSE program, allowing both depth-first or breadth-first searches of the tree when ambiguous responses are entered.

In the tree-building program, great emphasis must be placed on the provision of facilities to aid experts in entering their knowledge into the structure of a tree, in checking back over previous entries, making corrections, and making even major changes, all with ease and clarity. Emulating user access to the tree, plus effective use of graphics to inspect geographically the various parts of the tree and their relationships one to another, facilitate the checking of entered material both for errors of spelling and syntax, and for those of structure. The simpler errors can be easily corrected by the provision of full screen editing features, allowing insertion, deletion, and simple search facilities. In addition, powerful facilities for radical pruning and restructuring the tree are also essential. The construction and easy development of a word and phrase GLOSSARY, at any point during an emulation of user access, ensures appropriate and helpful entries.

The TREE TRAVERSE, or ACCESS program would allow a user to access the knowledge contained within the tree by eliciting responses to questions posed at every successive node, until finally some terminal position is reached. In the event that users may find that they cannot respond with certainty to a question presented at a given node, the system must present all possible terminations from that ambiguity onwards. The search method to be used to find these multiple solutions can be chosen to be either depth-first or breadth-first. In addition, the program must give a synopsis of the user's path through the structure, and allow a RETRACE if the users are unhappy with their previous decisions.

The hardware requirement

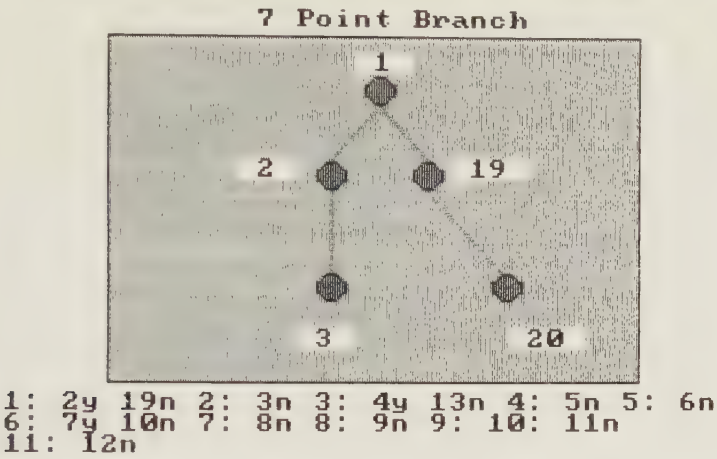
The target machine for the system was an IBM PC with 256 Kbytes of memory, a colour monitor, a pair of floppy disks, and a dot matrix printer. Several important sets of features of the IBM PC facilitate the construction, modification, development, and use of biological keys. First, the flexible colour properties and extended palette of the machine allow the production of easily-distinguished screen modes. For example, differently laid out screens with contrasting background colours can be used for data entry mode, edit mode, graphical tree display mode, and glossary mode, during the construction of the key. Similarly, clearly distinguishable tree traverse and glossary access modes can be available to the user when accessing the tree. Secondly, the graphics features of the IBM PC can allow the geography of the tree to be illustrated, enabling a constructor to orientate himself quickly within the tree, and hence move smoothly about the whole structure. Thirdly, the IBM PC's comprehensive string-handling facilities make it possible to provide features such as a full screen editor. Finally, the IBM PC Graphics Printer provides a series of alternative fonts and type sizes, which help the user produce easily followed tree listings, and synopses of chosen paths through the tree.

The Glasgow system

Tree building

The development of such a system, using an IBM PC, is the purpose of my recent work at the Glasgow College of Technology. It has been possible to achieve the facilities outlined above for entering, rearranging, and modifying expert knowledge.

The biggest of the programs helps the construction of dichotomous trees. To construct such a tree in any but the most obvious area is complex. The priorities of dichotomizing questions can be difficult to determine as few, if any, areas of study conform rigidly to a simple dichotomous hierarchy. On the other hand, a dichotomous tree *can* be constructed in spite of ambiguous priorities, by the assignment of arbitrary priorities to some nodes, while repeating others at different locations within the tree. Constructors of such a tree will generally start with a subsection of the overall area of knowledge with which they are most familiar. They might then develop work on several sub-trees at once, leaving



Enter next root

Fig. 7.1 The seven-point branch.

open the interrelationships between them until later. Subsequently, previously-constructed sections of the hierarchy might be considered to be incorrect, and consequently require drastic surgery and restructuring. Such tasks cannot be achieved at one sitting, nor can it be guaranteed that access to previously-completed work will find it entirely free of either major or minor errors.

Thus, this program, GENTREE, allows *partial entry* of the tree, and its saving on disk, with the possibility of later re-access for further additions or modifications. The program allows *editing* of any text entered into the tree, the *insertion* of new branches both within the body of the tree, *and* at the extremities, and finally the *switching* of whole substructures from one part of the tree to another.

Visualizing the tree

This latter requirement needs some visual aid in finding one's way about the tree, and this is supplied firstly by an algebraic string, listing every node number together with the numbers of its branches. These can then be displayed at the top of the screen at the touch of a single key.

The second and more powerful means available for 'knowing

where you are' allows the display of subsections of the tree using colour graphics. The form of the graphical display is a seven-point branch, which provides a picture of two levels of the tree emanating from the current node as ROOT. Further seven-point branches can then be called into view merely by entering the number of the new node required as ROOT (Fig. 7.1).

Editing and restructuring

Two major surgery options are available. One way is to take a REDIRECT option, which allows any branch responses and pointers, and following on with this new branch line further nodes can then be added.

Having broken into a sequence using redirect, the redundant sequence which originally continued on from the now redirected node, is, so to speak, hanging loose with no pointers to it from the main tree. In a sense, this sequence is itself a tree, with its own root, and it can be identified by using a LISTROOTS option which will list *all* such trees for later reintegration at an appropriate point elsewhere in the main tree.

The second method of restructuring is the MOVENODE option. This allows a given node to be moved to a new numerical position within the tree, automatically updating all pointers to this node wherever they occur. Aids to carrying out this change are available, giving, at the touch of a key, a list of all presently-used nodes. Having chosen a new number for the current node, the program makes all the necessary changes to pointers referring to it and then redisplay the list of nodes used.

More modest modifications are achieved by means of the built-in SCREEN EDITOR.

On completion of any entry of text into the tree, the program always responds with 'Text O.K?', allowing an immediate change if required. As many text entries can be quite long, a simple re-entering with correct text might prove onerous, and therefore, in addition to such a RE-ENTRY facility, the program also provides a SCREEN EDITOR with full control of cursor movement, and the ability to BACKSPACE (backwards deletion), DELETE (forwards deletion), and to INSERT at the current cursor position. To facilitate speedy modifications without excessive cursor movement, the program supplies a simple SEARCH mechanism, which positions the cursor on the first appearance of any given string.

Treefile - b:gras4.dat

Key for the Identification of Grasses in Lawns

1	T: Leaf-blades normally flat or rolled or folded lengthwise during dry weather - soft to firm but never bristle-like: Y: : N: : Leaf blades bristle like - usually tightly infolded or inrolled - narrowly grooved or channelled on the upper side - very narrow - usually hairless:	Pointers: 2 : 19
2	T: Ligule formed by a dense fringe of very short hairs - leaves bearded with spreading hairs at the junction of sheath and blade - blades 2-4 mm wide - blunt or abruptly pointed - folded in the shoot - afterwards flat or gutter-like - tufted perennial: Y: 'Heath Grass' - SIEGLINGIA DECUMBENS: N: Ligules membranous - variable in length - sometimes short and obscure:	Pointers: 0 : 3 :
3	T: Young leaf blade rolled in the shoot - the latter mostly cylindrical - circular in the transverse section - sheaths mostly rounded on the back: Y: : N: : Young leaf blade folded lengthwise about the middle nerve in the shoot - the latter sometimes more or less compressed - and the sheaths frequently keeled - especially in the upper part - rarely rounded on the back:	Pointers: 4 : 13
4	T: Leaves loosely to densely bearded with spreading hairs at the junction of sheath and blade - aromatic when bruised - green - ligules 1-5 mm long - blades 1.5-5 mm wide - finely pointed - tufted perennial: Y: 'Sweet Vernal Grass' - ANTHOXANTHUM ODOORATUM: N: Leaves not so bearded or aromatic when bruised:	Pointers: 0 : 5 :
5	T: Shoots usually swollen or bulbous at the base - blades 2-5 mm wide - finely pointed - green or greyish green - ligules 1-4 mm long - leaves hairless - loosely to compactly tufted perennial: Y: 'Smaller Cat's Tail' - PHLEUM NODOSUM: N: Shoots not thickened at the base:	Pointers: 0 : 6 :
6	T: Plants with whitish or brownish wiry rhizomes creeping through the soil and bearing brownish scale leaves - perennials - forming loose to dense tufts or patches: Y: : N: : Plants always without underground rhizomes - but sometimes with surface runners (stolons) bearing green leaves - otherwise loosely to densely tufted:	Pointers: 7 : 10

Fig. 7.2 Listing the tree.

The glossary facility

Another option is available whenever text has been entered. In the 'Addition to Glossary' facility, any words or phrases that might need explanation can be added to a list for later access during the running of the tree. On completing the session, the glossary is sorted automatically into alphabetical order before being saved on disk.

When accessing the tree, this facility allows users to enter any word on which they need further information, and the program will deliver (if available) any such text associated with that word by the builder of the tree.

Listing the tree

The methods of display of the structure of the tree outlined above leave something to be desired if major surgery is required in modifying the tree. Hence a program has been made available, with easy access via the main menu, to read a tree file from disk and print a hard copy.

The printed results display: the node number; the question text of the node; the 'YES' and 'NO' response texts; the 'YES' and 'NO' branch pointers (Fig. 7.2).

Maximum use is made of the alternative fonts available on the printer, and adequate space is provided for notes and corrections in the right-hand margin.

Thus, the user can take away this full tree listing to correct, annotate, and expand it for later updating of the tree via the main tree-generating program.

Accessing the tree

The previous programs eased the building of a dichotomous tree: the next program is used to access the knowledge embedded in the appropriate tree file by a user. Once again this option is available via the main menu, where the user is prompted to enter the filename of the tree file to be accessed.

Responding to a piece of text

Once the requested file has been loaded, a descriptive title is displayed on the console, and the first text from the root of the tree is presented to the user, who can respond by pressing:

(1) the 'Y' key—to take the 'yes' branch from this node;

- (2) the 'N' key—the 'no' branch.
- (3) the 'D' key—signifying 'don't know' or maybe.
- (4) the < spacebar >—which takes the user into the glossary option.

The first two options result in the appropriate 'yes' or 'no' response, followed by a move to the next node along the chosen branch, while the third option indicates that neither a positive 'yes' nor a 'no' is possible at this stage, in the opinion of the user. On being presented with a technical question, it is possible that a user will not be familiar with all the terms mentioned, and will require assistance. This is provided via the glossary facility which invites the user to enter the root of any word about which he or she requires further explanation (i.e. bristle rather than bristles, bristly, bristled, or bristle-like). The program will then respond with a stored explanation, if such exists within the author's glossary, before asking the question again.

The ambiguous response

When a 'don't know' response is entered, neither the left nor the right branch is indicated with certainty, so both must be explored. Also, additional ambiguous responses further down the tree will multiply the possible paths through the tree, resulting in a series of possible solutions. Each of these solutions must be presented to the user, and the sequence in which the tree is searched for these alternatives determines the speed with which an acceptable solution can be found.

Two search patterns are available. The first is a *breadth-first* search which is generally agreed to be most appropriate when there are a small number of possible solutions—the likely case if we are constructing a biological key. The second is a *depth-first* search, which, though less efficient in speed, has the advantage of sequencing through all closely-related terminations to the tree one after the other, rather than leaping from branch-to-branch in a seemingly unrelated way as is the case in breadth-first searching.

Several features are included to facilitate the handling of ambiguous responses by the system, or to help the user.

For example, whenever a 'don't know' is entered at a particular node in response to the displayed text, the system further assists the user by offering the *negative* response (which would normally only have been displayed as a response to a 'no' entry) as a question to which the user must reply. Thus, both branches of the tree at the

node will have been presented to the users to facilitate their choices, and perhaps to overcome any ambiguity connected with the 'yes' branch, by the need of a firm response to the 'no' branch.

It is only when *both* options have been responded to with a 'don't know', that *both* branches are searched. The sequence is to take the 'yes' branch first, after saving the current node number. Later on, backtracking to this node is achieved by reference to the stored node number, followed by a search of the 'no' branch.

Some further points should be made here about how ambiguity can be overcome and a single solution discovered.

One of the features of well-structured dichotomous trees is that a wrong turning at a particular node can be revealed by responses to subsequent questions further along the path. Cases occur when *both* branches are obviously wrong and are rejected, and this can be used to trigger the abandonment of the current path, followed by backtracking to the next available saved ambiguous node. Erroneous routes through the tree can therefore be identified, to

Key for the Identification of Grasses in Lawns

NOTES

Leaf-blades normally flat or rolled or folded lengthwise during dry weather soft to firm but never
er bristle-like

N - Leaf blades bristle like - usually tightly infolde
d or inrolled - narrowly grooved or channelled on the up
per side - very narrow - usually hairless

Roots cord like - shoots closely packed on short rhizomes forming dense tufts - basal sheaths tough
and shining - ligules about 1 mm long - blades sharply pointed - rigid

N - Roots very slender - basal sheaths not hard and shinin
g

Ligules extremely short and obscure - less than 0.5 mm long

Y -

Plants densely tufted - without rhizomes

N - Plants with slender creeping whitish or brownish scaly
rhizomes - forming loose to dense turf - leaf blades 0.5-
1 mm wide - dark green or greyish green - basal sheaths ha
iry or hairless - 'Red or Creeping Fescue' FESTUCA RUBRA
A RUBRA

Fig. 7.3 Hard-copy synopsis.

arrive at a solution in spite of ambiguous responses at a series of nodes.

When ambiguities cannot be resolved as outlined above, the system will deliver all possible solutions in sequence, the actual order of the solutions being determined by the type of search method selected.

The presentation of solutions

A synopsis of all user-affirmed statements is listed on the screen, enabling the users to inspect their chosen paths and consider whether there is any doubt about any of their decisions. If a re-run of the tree is likely to be required, it would be useful to have a hard copy of the synopsis suitably expanded, to list not only the chosen responses, but also the alternatives, and this option is made available at this point in the program. These printed results can be corrected and annotated, to facilitate the right choices when the tree is run again (Fig. 7.3).

Now, if the situation is one where there have been any ambiguous responses, the two synopsis options above are modified to indicate that each solution is only one of a series. After each, a return is made to process subsequent alternative solutions commencing with one of the previous saved nodes.

Re-running the tree

Finally a re-run option (or RETRACE as it is called in the program) is offered, with precise information as to how the user chose previously, and with the possibility of standing by a choice or changing it.

If, during a re-run, a change is implemented at a particular node, the whole re-run option with its details about previous decisions becomes irrelevant and is abandoned. Thereafter, the normal tree accessing is resumed until a new terminal node is reached.

Conclusions

The software described here has been found to be particularly useful to biologists at Glasgow College of Technology, who need to classify organisms new to science within an (at present) inadequate taxonomic framework (in this case Tardigrades), and thereafter to construct and use biological keys for identification purposes. Development is currently taking place on extending the above facilities to lateral tree structures.

**JIM SCHOFIELD**

Jim Schofield has spent the last 16 years in higher education, latterly in Hong Kong and Scotland. He is at present Principal Lecturer in charge of the Computer Services Unit at the Bedford College of Higher Education. After a period working on system software, his main area of work for the past three years has been concerned with the application of microcomputers in the laboratory. Automated control of experiments in chemistry, biology, and mechanical engineering, have led to an interest in expert systems, particularly in the area of dichotomous and lateral trees.

8 Using the IBM PC in the school classroom

B. MILLO

Informatics Education Unit (HSTP)
Dept of Education
University of Southampton
Southampton, Hants
SO9 5NH

Introduction

In recent years computers have become essential items of equipment in British schools, to the extent that the Government has provided funds to make sure that every school in the country is equipped with some computer power.

The purpose to which the machines should be put has not, however, always been terribly clear; and there are still many teachers who feel guilty because they have not yet used the school's computer in their own classrooms. Much special software has been written for use on the Acorn BBC Microcomputer and the Research Machines computers: software that might be termed 'educational', or 'content specific', in that it is designed with a narrow and specific purpose in mind.

Examples of such software are 'Mary Rose' (1), submarine excavation in the Solent; 'Micro Mapping' (2), developing Atlas skills; and 'Angles and Bearings' (3), developing geometrical skills.

The Hampshire Project

In 1984 plans were laid for the establishment of a project, to be called the Hampshire Software Testbed Project (HSTP), which would look in a different way at the possible uses of software in some Hampshire schools.

The question to be asked was: 'If standard commercial software is

provided to teachers and students in the classroom, what uses do they find for it?' In this context, it was thought that the kind of software to be used would fall into the three categories of word processors, database handlers, and spreadsheet handlers.

The project was to run in Hampshire, including mainstream and special schools, and would be supported by grants, hardware, and expertise from IBM United Kingdom Trust, Hampshire County Council, and Southampton University. As part of the exercise was to explore the uses of *commercial* software, it was important to ensure that suitable computers (on which the software would run) were available. To this end, IBM gave to the project 50 of its Personal Computers, initially configured with two single-sided disk drives and 128 Kbytes of main storage.

The mainstream schools that were asked to participate comprised two sixth form colleges, two secondary schools and two primary schools; these were joined later by three special schools, catering for children with physical, mental, and social disabilities.

The early observations were somewhat as might have been expected:

1. Word processors have obvious applications in the classroom.
2. Junior school children do not find much use for complicated spreadsheet handlers.
3. In general, children tend to learn techniques of use more quickly than their teachers.
4. A great deal of time is needed for teachers to become confident users of commercial software.

The last of these points has been keenly felt, in that it was not until the second year of the project that participating staff really began to feel at home with the computers themselves and the software. However, as the project progressed through its second year, ideas for classroom use began to be developed, and some of these ideas have been printed in a series of monographs. The remainder of this article is devoted to describing briefly the content of some of these.

A simple application of database handlers

Personal assessment

Some of our 'difficult' students are required to assess their own personal characteristics (honesty, fairness, politeness, etc.) in

MY PERSONAL RECORD:			
MONTH NUMBER:			
SCORE:		SCORE:	
Honesty	:	Effort	:
Kindness	:	Fairness	:
Politeness	:	Good Temper	:
Helpfulness	:	Friendliness	:
Reliability	:		:
Cheerfulness	:		:

File: b:fred.w	FORM 1	Page 1
F2-Print	F5-Date	F6-Time
		F10-Continue

Fig. 8.1 Layout of the database for pupil's self-assessment.

consultation with their teachers. This can be a difficult task, but it is a valuable one for the children.

A teacher from one of our special schools felt that pfs:File and IBM's Graphing Assistant could be of value in this exercise, and devised a database in which the children record their personal assessments. This in itself would probably not be a great improvement over the usual paper and pencil activity, but the ability to go on and have successive months' assessments illustrated automatically on a graph is a great incentive to the children, in both completing the exercise and improving their behaviour.

Figure 8.1 shows the layout of the database record on which each child records (in consultation with the teacher) a personal assessment of behaviour for the current month. The numbers to be recorded would typically be the subject of much discussion, and perhaps disagreement, between the two, but would eventually represent an agreed view of the child's behaviour. The child can then proceed to the next stage of the process, which is to use the graphing program to pick up the values from the database records, and show comparisons between different characteristics of personality (see Fig. 8.2).

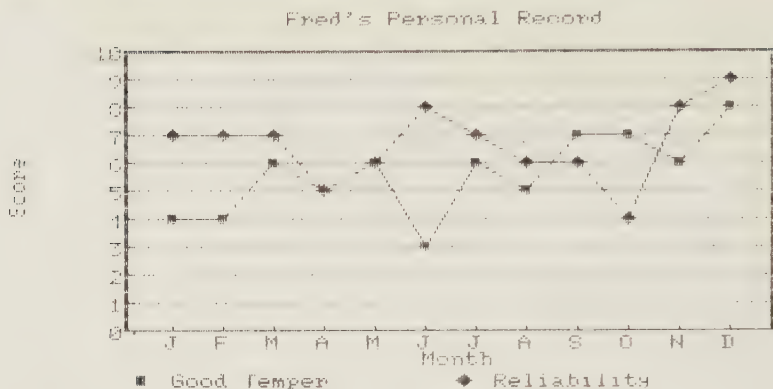


Fig. 8.2 Graph of pupil's self-assessment, using the pfs:File database and Graphing Assistant.

A secondary advantage of this approach is that the children using it gain some experience of the use of a graph to illustrate sets of values. The fact that they do not have to struggle with scales and the plotting of points (both of which are done automatically by the program) means that they can give their full attention to understanding the implications of the data that they have recorded.

More advanced applications of database handlers

Cataloguing a school's resources

One of the major difficulties in a large school is the tracking down of information of all kinds: not only books for reference, but also slide sets, videotapes, and so on. Moreover, this is an activity that is increasingly being undertaken by the children rather than the staff, as part of the children's education.

Giving the children access to an up-to-date database of such information fulfils two functions: it makes the information more readily accessible when it is required, and it gives the students the opportunity to use the computer in a real, rather than a contrived, situation.

There have been a number of attempts to do this, and there is nothing original about this project's work in this respect, but the successful employment of the large and powerful package Datamaster is, nevertheless, a worthwhile achievement.

SMILE activities

The SMILE programme (Secondary Mathematics Individual Learning Experience) has been devised as a set of work activities that are used according to an individual programme that is tailored to an individual student.

The monitoring and control of these activities is a time-consuming business, usually taken on by the teacher, but if they are recorded on a sufficiently fast and capacious database then the administrative aspects become relatively trivial, and the teacher is released for the more important teaching that needs to be done. Meanwhile, the child carries out the recording of results and the searching for new activities, again enhancing the whole educational experience.

Applications of spreadsheet handlers

Census analysis

A quantity of numerical information relating to populations, generally referred to as census data, has been available for many years. The transfer of some of this data to a sufficiently powerful spreadsheet handler can bring to life statistical aspects of the data, especially if a suitable graphing program is also available.

Such data, from Portsmouth for the years of 1661 to 1680, has been transferred to Multiplan, so that the mathematical features of the software can be used to bring out some of the remarkable characteristics of the period.

Periodic Table of the elements

Mendeleev's Periodic Table is a valuable and necessary tool for the proper understanding of chemistry, but some of its implications cannot be seen or grasped unless the table is packed with information. After all, not only must the elements be grouped according to outer shell electron counts, but the atomic weights, melting points, boiling points, ionization energies, and so on, must be indicated as well.

An electronic spreadsheet that is sufficiently large can be set up with all this information, rather in the way that a written table can be built with all the values entered by hand. The advantage of the electronic spreadsheet is that, given sufficient power in the software, cross sections can be picked out and patterns observed, in a way that is not possible on a hand-drawn table.

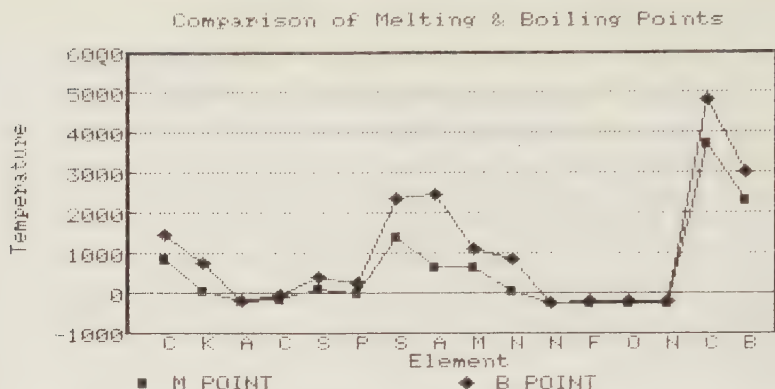


Fig. 8.3 Graphical comparison of the melting points and boiling points of elements, using data taken from the Periodic Table spreadsheet.

If a graphing program is available that can also read the spreadsheet data, then the students can be given the very real opportunity to discover for themselves the properties that become apparent from studying the Periodic Table.

Figure 8.3 shows a simple comparison being made between the melting and boiling points of sixteen elements. Additional value can be gained from the exercise by discussing in class the difficulties of using a program that: (1) automatically abbreviates names on the abscissa (the symbolic names for the elements in this case), and (2) automatically sets a scale for the ordinate that extends to -1000 degrees Celsius!

This exercise has been carried out, but it must be said that for ease of illustration an integrated package (such as Lotus 1-2-3, which has not been generally available to schools within the project) is preferable to the use of separate spreadsheet and graphing packages. This is because the time and expertise needed to switch from one part of the system to another should not dominate the real work that is to be done, namely, observing correspondences within the table itself.

A general application

Magazine production

One of the more important aspects of education generally, and of the

education of children with special needs, in particular, is the encouragement of them to work together in true co-operation. This can be achieved by getting them to produce a school or class magazine, with different children producing different parts of the finished product.

Again, this is not a new idea, but if the children have access to software that is easy to use but still powerful enough to enable them to produce high-quality results, then a great advance over the traditional methods is made.

We have found that when discussion takes place at the keyboard between two people who disagree over the content of a particular report for the magazine, they learn the need for co-operation and compromise and, more particularly, they learn how to achieve it. Whereas the time taken over such disagreement might result only in impasse under the paper and pencil approach, the computer allows the protagonists to adjust their copy continually, and even to record similar yet different versions for later discussion and resolution.

Even the most disabled of children can produce work that all agree is of value, because they can so easily make various trials of the work they are attempting until a suitable version is reached. This is especially true of children who want to draw particular pictures and who, without the computer's capacity to permit continual correction and refinement, only become disheartened by the failure of their muscles to respond to their wishes.

We have made available to quite young children a word processor, a drawing program, and a caption writer, with the result that they have been able to produce a complete magazine.

Conclusions

The Hampshire Software Testbed Project was not planned as a full research project, with detailed planning and evaluation. It was considered important not to oblige teachers to participate and to 'come up with results', but rather to make available to them a variety of resources that they would not normally have, and to provide them with what they wanted, when they wanted it.

As a result, we cannot say why a particular piece of software will work in a particular situation, or why it won't; only that we have found it to be so. For example, in the first term of the Project, 10-year-old girls could be observed making use of Peachtext to compose text for a class newspaper; they were coping well with its

intricacies, finding out on their own initiative, for example, how to adjust column widths to suit their needs. Most of the other schools, however, felt that this particular package was too complicated to be of much use, and quickly discarded it in favour of simpler word processors.

Again, Multiplan can be a daunting program to get to know, and yet Junior children with a particular application to model were able to control enough of its extensive function to make it an effective tool for their own work.

It may be that the children who were observed to cope well with complicated software were particularly bright; on the other hand, it may simply be that when the children themselves perceive a need to solve a problem of their own, they pursue potential solutions with a degree of determination that leads to success.

One of the problems with which the project had to cope was that of training the staff to a point at which they felt at ease with the software and hardware that had been made available to them. It is hardly appropriate, and certainly not customary, for teachers to attempt to make use of material with which they are not fully conversant and whose value is not clearly and confidently understood; but it is difficult to justify the expenditure of significant amounts of time in reaching the level of confidence in items of software whose price is high and whose very content is continually shifting as new products are published.

It has, however, become clear to the author, as manager of HSTP, that in cases where a teacher can encourage the children to experiment with software (while the teacher perhaps remains in ignorance of the details of the packages), the children themselves will find useful applications and will achieve a degree of critical familiarity that can only improve the future situation.

Our report (available upon request from the Education Department of the Hampshire County Council) describes the effects that have been observed, and will, we hope, encourage others to take our findings and move on from there with their own experiments and observations.

References (software)

1. *Mary Rose* is published by Ginn & Co. Ltd, Prebendal House, Parson's Fee, Aylesbury, Bucks, HP20 2QZ.
2. *Micro Mapping* is published by Thomas Nelson & Sons Ltd, Nelson

House, Mayfield Rd, Walton-on-Thames, Surrey, KT12 5PL.

3. *Angles and Bearings* is also published by Thomas Nelson & Sons Ltd.

BRIAN MILLO



Brian Millo was educated at the City of London School and at Clare College, Cambridge, where he read mathematics. After graduating he taught mathematics, physics, and computing for seven years, first in Hong Kong and then in London, before joining IBM in 1971 as a PL/I programmer on a large internal application.

In 1973 he transferred to an internal education department where he remained for some years and became its chief instructor. During that time he taught most components of the program development process, eventually specializing in modern techniques of program design, and in the inspection process.

Following a short spell in quality and productivity, he left IBM and set up his own training company, offering his expertise and experience to the wider world of data processing. For the past two years a significant proportion of his time has been taken up with running the project referenced in this article, in nine Hampshire schools. Brian is a Member of the Institute of Data Processing Management.

9 Logo and the IBM PC in action

M. THORNE

*Department of Computing Mathematics
Mathematics Institute
University College
Senghennydd Rd
Cardiff, CF2 4AG*

Introduction

Logo is not just a computer language. Rather, it is an educational philosophy in which debugging, that process of finding and eliminating errors which is so germane to programming, is regarded as the most powerful educational aspect of program writing. The more familiar red ticks and crosses, however dutifully inserted by the teacher, are hardly an incentive to correct what has gone wrong. Who can fail to remember the galling experience of 'being made to do your corrections'? On the other hand, programming allows the student to get it wrong. No one in their right mind expects a computer program to work first time!

The other major aspect of Logo philosophy centres on giving children as much control over their own learning as possible. First, the language itself offers even young children access to the complicated mathematical idea of a differential equation. Using simple commands like FORWARD 10, LEFT 20, and BACK 9, they program first a toy turtle which runs along the ground, and then a pretend turtle which moves about the computer's screen. A pen attached to the turtle's tail leaves a trail on the floor or computer screen as it moves. In this way, five-, six- and seven-year-olds are given access to ideas in differential geometry, previously the privileged domain of third-year mathematics undergraduates.

Secondly, because children are more committed to, and become better involved in, projects of their own making, good use of Logo

requires that the *children* decide what to do. This is not a return to the wishy-washy educational *laissez-faire* of a decade ago as research has now turned in favour of the idea that children need gentle nudges in the right direction. Logo is a superb descriptive tool, as relevant and important to children's mathematical development as drawing or painting, and good use of Logo is akin to the teacher choosing a subject for painting and leaving the children free to respond to the subject.

Children using Logo

Children begin by trying to understand the Forward, Back, Left and Right behaviour of the screen turtle in conjunction with a mental image of some real-world object—possibly an actual turtle, or a bulldozer, or themselves—to which they can refer when the going gets tough. Hence, 'walking through' turtle paths becomes an important technique for debugging and constructing.

Turning requires special attention: most important of all, the children have to realize that it does not cause anything to be drawn. The difficult part for the teacher is to help them become aware that Left and Right are inverses, and that turns of 90 and 180 degrees are special—without actually making the discovery for them.

Combinations of turns and Forward and Back are then explored for both non-closed paths and polygons. Again, real-world equivalents are important; young children are still very much 'concrete' thinkers. There is a nice two-player game which can be used here; the players take it in turns to go, Back is not allowed and the object is to be the player who gets the turtle back to its starting point using only Forward, Right, and Left.

The special closed paths forming a square and a rectangle are, of course, important discoveries, and these may then be used as building blocks for very simple pictures: e.g. two squares at once. Slowly, some children may become aware of the need for, and utility of, sets of instructions known under one name, and the difference between the name 'apple pie recipe' and the instructions for making it.

This motivates the Logo procedure concept. A collection of instructions in Logo is formed into a procedure by adding the keyword TO and a procedure name at the beginning, and the keyword END at the end. Hence, a procedure to draw a square could be defined as:


```
TO SQUARE  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
END
```

Thus, although a Logo program to draw a pin person might look like:

```
TO PINPERSON  
FORWARD 300 RIGHT 120 FORWARD 300 RIGHT 120  
FORWARD 300 LEFT 120 FORWARD 300 LEFT 120  
FORWARD 300 RIGHT 180 FORWARD 300 RIGHT 120  
FORWARD 300 RIGHT 180 FORWARD 300 LEFT 120  
FORWARD 150 LEFT 45 FORWARD 50 RIGHT 90  
FORWARD 50 RIGHT 90 FORWARD 50 RIGHT 90  
FORWARD 50  
END
```

it is very easy to make errors, and an error is extremely difficult to locate. The teacher must nudge the child in the direction of using subprograms so that the task can be broken down. Thus:

```
TO PINPERSON  
LIMBS  
FORWARD 50  
LIMBS  
FORWARD 25  
HEAD  
END  
  
TO LIMBS  
RIGHT 120  
FORWARD 50  
BACK 50  
RIGHT 120  
FORWARD 50  
BACK 50  
RIGHT 120  
END
```

```
TO HEAD  
RIGHT 45  
REPEAT 4 [FORWARD 20 RIGHT 90]  
END
```

is of a modular construction and so much easier to test and to understand.

It is important to emphasize that there is no fixed stage at which certain concepts in Logo must be discussed. The art is in restraining oneself to providing only those concepts for which the children are genuinely in need. For instance, variables might only be mentioned when they ask if there is a way of drawing squares of different sizes without defining a new procedure for each of the different size squares they want to draw. (Professional computer scientists might like to compare this with the perceived wisdom that real programmers normally use only a small percentage of the total facilities available in a given programming language.)

Once again, real-world images of the variable concept are crucial to progress. Thus, variables give you lots of things from one source—given one video recorder you can see lots of different films; variables bring lots of different things together if you can find one or more factors which express the difference between these things—for example, three different cars are actually 2-, 3-, and 5-door models of the same car; lastly, a variable may have a current setting—the speed of a gramophone, the colours of the traffic lights.

Higher order skills, such as procedures which call other procedures and recursion, may only be used by some of the children but real-world examples are all around them—think of the leaning tower of Pisa and Japanese houses. Choosing the ‘easiest’ way of going about a programming task (e.g. tile the whole screen with hexagons) gives a major focus to classroom problem-solving activity. A great deal of research has been done and is continuing into understanding children’s problem-solving styles, abilities, and weaknesses, and Logo has an important role to play in that work.

List-processing work has so far been undertaken mainly by teachers in research projects, but this may be due to a lack of time for teacher training rather than problems with the list concept itself. Certainly, lists are not easy but they do have a natural place in school. For example:

- (1) anyone not on the exam list, see me after assembly;
- (2) here is the list of the cricket XI to play Next Street Comprehensive on Saturday, the last two names are the substitutes;

- (3) all but the first five must see Mr Bloggs for extra homework;
- (4) all but the first on each form list will come straight into the hall after first period, the others will take registers to the secretary.

These model both the Logo list concept and the available list processing operations FIRST and BUTFIRST.

The Microworlds project

The Microworlds project is a scheme which hopes to provide pupils and teachers with both a book and a series of video tapes, to help them learn Logo. So far, the book has been published and the video scripts written.

One of the main aims of those involved in the Microworlds project has been to give serious consideration to the idea of a 'Logo culture'. On one level, of course, the provision of far more numerous and powerful machines will contribute greatly to the creation of such a culture (at the very least it will allow children themselves to take over some of the roles at present undertaken by teachers). But there is more to it than that.

Children do not learn to speak without intervention from the adult world. They may learn without teachers, but they do not learn without teaching. Everywhere they are surrounded by people talking to them and to each other. They communicate with adults, watch television, read books (or have books read to them). Their environment is full of signposts, advertisements, and street signs. In short they are surrounded by the *culture* of language.

The challenge is to build a Logo culture which has the genuine appeal of, say, the 'Space Invader' type of computer culture. It is unlikely that such a culture will be built by accident. And given the existing lack of such a culture, it is hardly surprising that children do not automatically think in a Logo-orientated way. Where is a child likely to pick up the idea of modularity, or even of elementary heuristics such as 'try a simpler problem'? Let alone variables and recursion . . .

The book *Microworlds: Adventures in Logo* (Noss *et al.* 1985) is a unique collection of exercises and ideas devoted to programming in Logo. Underlying the whole of Microworlds is the fundamental idea of allowing children an opportunity to control their own learning through the Logo language. This means that the contents are far from the didactic teaching approach associated with most teaching

of programming. Instead, each double page contains material to plant seeds in children's minds; to provide counselling when the going has got tough with their own computer programs; to license free exploration and getting it wrong; and to help children find long-term goals in their Logo work. The main theme of the *Microworlds* book is that Logo is a superb tool for modelling, i.e. creating descriptions in Logo of processes the children see around them. Illustrations of modelling are provided in many different environments and the subject areas include architecture, biology, botany, linguistics, and psychology.

Earlier spreads in the book offer children games to play pretending they are turtles. There are activities with a real 'turtle' (a toy would do for some of them). Recipes are used to introduce the idea of a program—the children can bake the turtle's favourite biscuits. Gradually, all commands of Logo are covered but the teacher is left a great deal of freedom to guide young learners on to sections about Logo commands of which the children have a need. To stimulate the desire to use more commands, complete programs are also given which the children are not expected to understand in advance. These include a Logo fruit machine and a pop-up spook. Hopefully they will motivate children to make their own improvements by amending and adding to the given programs.

Many pages could form the basis of a class project, allowing each child to contribute programs at his or her own stage of Logo maturity. For example, one spread raises the idea of a Logo identikit set: one group of children could provide eyes, another noses, and so on.

The final pages in the book offer ideas about processing lists of things, like all the people you would like to come to your birthday party. Here, and elsewhere, fundamental ideas from more traditional mathematics are interlocked with Logo: changing all your LEFT turns to RIGHT and vice versa in a Logo program illustrates one of the fundamental ideas of transformations.

People often ask: what is it children are doing when they write Logo programs? Mathematics is certainly involved: problem solving, geometry, modelling, and arithmetic, are fundamental. But, as anyone who works through *Microworlds: Adventures in Logo* will discover, there's a whole new culture waiting for them.

Yet the book is just stage one. For stage two we have scripted five television programmes aimed at making the four children—Orcim

and Ogol from planet K64, Jeff and Jane from planet earth—as vital a part of children's culture as popular cartoon characters.

Logo versions on the IBM PC

This work was developed on a wide variety of machines using, principally, the LCSI (Logo Computer Systems Inc.) version of the Logo language. More recently I have adopted the Harvard Associates version of Logo. It comes with an excellent manual which children can understand (and enjoy!) and a suite of demo programs. One of these is an implementation of the dynaturtle, a turtle which is free to wander around space where, of course, there is no gravity.

Using the K(kick), L(left), and R(right) keys, the dynaturtle controller must use Newton's laws of motion to keep the turtle on a race track. Sounds easy? Buy the system and try it out with your own children (or on your own if you're easily embarrassed). You'll very soon be convinced of the educational potential of such simulations.

Future directions

In several schools in South Wales, work is now going on which involves a version of Logo which can be used to control hardware devices connected to the computer via an interface box see Fig. 9.1. The children have built several devices, including burglar alarms with several different sensors, and a miniature production line conveyor belt, as well as carrying out some interesting creative work

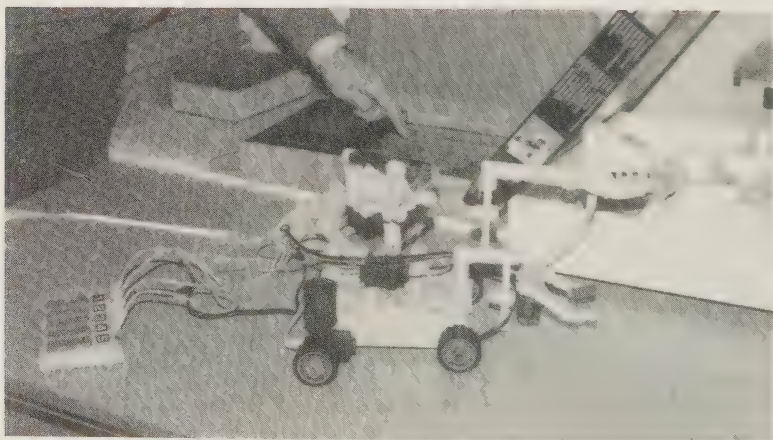


Fig. 9.1 A turtle, built and controlled by the children themselves.

based around the Robotix kits from Milton Bradley. By replacing the supplied controls by connections to the computer, these toy monsters can be brought to life and made to perform tasks like climbing stairs. (The educational value of such an activity is readily apparent when you watch the children involved.) Unfortunately this is not yet possible with IBM PC versions of Logo.

A South American researcher has recently been experimenting with three-dimensional Logo. Extra routines are supplied as ordinary Logo procedures so that any version of Logo can use them. These routines enable the turtle to be directed into the screen, so to speak, and may give teachers considerable insight into a child's view of spatial relationships.

Work is also going on in London University using sprite-based Logo. This allows multiple turtles, so that each Logo command must also specify to which turtle it refers. Sprites bring the concepts of parallel operations and motion into the Logo arena. With Sprite Logo it is even more natural to think of getting children to create screen-based animations—an advert, a play, a simulation of a space flight—which is where the next generation of child-orientated, Logo-like languages seems set to focus.

Reference

Noss, R., Smellman, C., and Thorne, M. (1985). *Microworlds: Adventures in Logo*. Hutchinson Educational, London.



MICHAEL THORNE

Michael Thorne is a lecturer in the Computing Mathematics Department at University College, Cardiff. He gained a first-class degree in mathematics at the University of London, and a Ph. D. in Computational Group Theory

from the University of Birmingham for his work on the 'Monster' simple group. The group has 10^{56} elements and its smallest representation with complex numbers is in 196 883-dimensional space. He continues to be active in this field, having recently, with a research student, produced a system for an IBM PC which will calculate in such a group.

Over the past seven years he has become widely known in the field of computer-assisted learning, until recently being the consulting editor for *Applied Systems Knowledge*. He has taught at every level from primary school to university and contributes to *The Times Educational Supplement* and *The Listener*.

As one of the presenter of ITV's 'Database' programmes and Channel Four's '4 Computer Buffs', he is also well-known among computer hobbyists.

10 Integrating the BBC Microcomputer and the IBM PC

L. BURBRIDGE and P. BUTTNER

*Computer Centre
University of Bristol
University Walk
Bristol, BS8 1TW*

Introduction

In common with many other UK universities, the University of Bristol has a large number of BBC Microcomputers and a significant number of IBM PC XT and AT systems. As at January 1987, there were approximately 800 BBC Microcomputers and about 250 IBM PCs installed. In the past a number of departments have invested heavily in BBC Microcomputers. However, the advantages in moving towards the IBM PC are now becoming apparent. For administrative purposes, such as word processing, information storage and retrieval, accounting, etc., the wealth of high-quality software available for the IBM PC is a considerable advantage and the need has emerged for an upgrade path. Similarly, the IBM PC AT is a system well-suited to many scientific applications, particularly when used with Fortran 77. Academic staff and research students are likely to find that both BBC Microcomputers and IBM PCs are available within a single department or research group. With the above background, it is clear that there is a requirement for compatibility between these two very different systems. The compatibility involves two distinct factors:

- (1) a method of transferring text and data files between the BBC and the IBM PC;
- (2) a facility to allow users to take BBC programs and to run them, unchanged, on an IBM PC, (these programs will invariably be written in BBC BASIC).

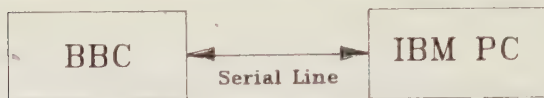


Fig. 10.1 Direct serial connection.

The following sections describe the methods by which the above requirements can be met.

File transfer

In general, there are four methods of transferring files between a BBC Microcomputer and an IBM PC; these are shown diagrammatically in Fig. 10.1–10.4 and each method is discussed below.

Direct serial connection

It is a relatively simple matter to connect a BBC Microcomputer to an IBM PC via a serial line (Fig. 10.1). A file transfer program can then be used to transfer files in either direction. Some form of error detection and re-transmission is essential and a version of Kermit, a file transfer protocol developed at Columbia University in the USA, is now available on most systems to provide this. The method is cumbersome, the two systems must be in relatively close proximity, and both machines must be dedicated to running the file transfer program during data transfer. This method is also relatively slow. Many users expect the data transfer to take place at the nominal speed of the serial communications link, which is typically 9600 baud (bits per second). Due to the required error checking and other factors, the effective data transfer rate may be much less than the nominal rate. Thus the data may only be transferred at, say, half the nominal rate. This will result in a transfer rate of about 600 bytes per second, or about 10 minutes for a full 360 kbyte floppy disk.

Serial connection via a host computer

This method assumes that both the BBC Microcomputer and the IBM PC have a serial connection to a suitable host computer (Fig. 10.2). As before, Kermit can be used to transfer the files, as most host computers now offer a Kermit facility. In effect, the host acts as a data buffer between the two microcomputers. This provides some flexibility, since the BBC Microcomputer and the IBM PC

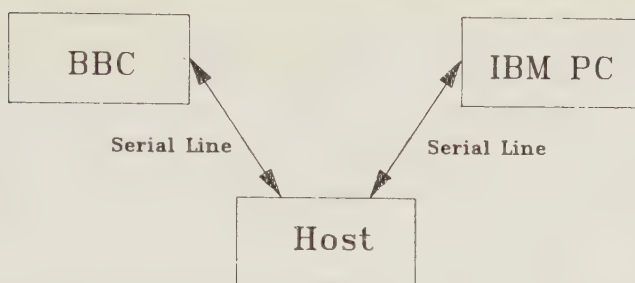


Fig. 10.2 Serial connection via a host computer.

need not be available concurrently, and the systems may be located in different geographical locations. The major disadvantage is the unnecessary use of host computer resources. As with the first method, typically serial line data transfers will be limited in speed. In this case, however, the data transfer will take twice as long since the data must be transferred twice.

Reading and writing IBM PC-compatible disks on a BBC

In the case of the BBC B Plus or BBC Master, equipped with a double-sided 40-track 5.25" disk drive, it is possible to read and write IBM PC-compatible disks directly (Fig. 10.3). This method requires suitable software. One such package, which works well, is DOSCOPY (available from Baksoft Ltd, 20 Leys Avenue, Cambridge). In effect, any BBC disk file can be copied directly to or from an IBM PC disk using only the BBC disk drive(s). The IBM PC-compatible disk may then be removed and placed in an IBM PC system in the usual way. This method is cheap, effective, and fairly convenient. The disadvantage is that it is not possible on the large number of earlier vintage BBC Microcomputers. It is, however, one of the methods we recommend of providing a link between the two systems.

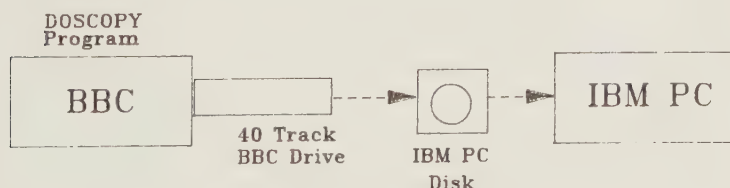


Fig. 10.3 Reading and writing IBM PC-compatible disks on a BBC.

Reading BBC disks on an IBM PC disk drive

The ideal method of using BBC files on an IBM PC is to be able to take a standard 40-track 5.25" BBC disk, place it in the IBM PC disk drive, and read it directly (Fig. 10.4). This apparently simple operation does, however, require both a hardware modification to the disk controller card of the IBM PC and the development of suitable software. The actual method employed is discussed in the appendix. Although this method clearly requires some expenditure and has also required a substantial investment in software, the advantages of this system are sufficient to justify the expenditure. Users are then free to produce files on a conveniently-situated BBC Microcomputer (often at home) and then use the same disks on, say, a departmental IBM PC system.

During 1985, a prototype card constructed by one of the authors was installed, together with the necessary software, in a user department. The system proved to be successful and a few additional cards were constructed. In order to preserve the warranty on the IBM PC it was, of course, necessary to purchase replacement disk controller cards which could be freely modified. Clearly, such individually-constructed cards were not a practical proposition for meeting the needs of an increasing number of users. It is pleasing to report that in early 1987 the problem was solved when the 'Compaticard', manufactured by Micro Solutions of Illinois, was released on the market. This card provides a commercial solution to the problem. The cost is relatively modest compared to the high level of utility provided to BBC/IBM PC users. (The card is available in the UK from The Core Store, 59 Station Road, Northwick, Cheshire.)

BBC Basic on the IBM PC

BBC Basic is a superset of Basic and contains a number of extensions which are not available in other versions of the language. These



Fig. 10.4 Reading BBC disks on an IBM PC disk drive.

extensions include named procedures with parameter passing, good graphics facilities, the ability to include in-line assembly language routines, and a very wide range of utilities. The consequence of these extensions is that a user who has developed programs in BBC Basic will have major problems in converting such programs to run on an IBM PC under Basica; in many cases, a complete re-write will be required. Clearly, the possibility of running unchanged BBC Basic programs on an IBM PC is very attractive and will be of considerable benefit to users migrating from the BBC Microcomputer to the IBM PC. Fortunately, this is now possible, thanks to a software package called BBC Basic (8086) developed and marketed by M-TEC of Reepham, Norfolk. This excellent program supports nearly all of the features of BBC Basic, including procedures and graphics commands, and very many programs will run completely unchanged (albeit slower!) on an IBM PC. Thus, the combination of this program and one of the data transfer methods discussed above, provides a simple and straightforward method for users to combine the use of BBC Microcomputers and IBM PCs. A few comments are necessary for intending users. The present version of BBC Basic for the IBM PC supports only the CGA graphics system and cannot be used with the Hercules monochrome graphics card; neither can programs take advantage of the full capabilities of the Enhanced Graphics Adapter and Display. The range of vertical coordinates is also different and all graphs must therefore be re-scaled. It is understood that versions of BBC Basic with support for the Hercules monochrome graphics card and the EGA system are under development and are due for release in the near future.

Other software issues

Apart from BBC Basic programs, it is helpful to be able to transfer other files. Two particular cases are frequently encountered. First, many users use the View word-processing program on the BBC Microcomputer. It is a relatively simple matter to write a utility to convert View files to either ASCII text files or to a format compatible with Microsoft Word or Wordstar. Typically, it is only necessary to translate the special characters used for soft spaces, soft carriage returns, boldening, and underlining. Other, special, characters are best translated or removed manually. Secondly, it has been found helpful to be able to transfer data from various BBC Microcomputer programs to Lotus 1-2-3 worksheets. Here, the

usefulness of the Lotus 1-2-3 FILE/IMPORT/NUMBERS function becomes apparent. All that is required is for cell entries to be separated by commas or by spaces with a carriage return/line feed sequence at the end of each row. This is, of course, the format of a typical ASCII text file; most programs have the capability of writing such output to a disk file. (As with many other programs of this type, cell entries containing commas must be enclosed in quotes.) Thus, it can be seen that it is possible to transfer View word-processing files to most word processing packages running on the IBM PC, and the output from most programs running on the BBC Microcomputer can be read into a Lotus 1-2-3 spreadsheet on an IBM PC. This, together with the ability to read BBC disks on the IBM PC (using the method referred to above), completes the integration of these two very different systems.

Conclusions

Using different and incompatible microcomputer systems within the same department or institution always presents problems for users. In the case of BBC Microcomputers and IBM PCs, however, these problems can be minimized to the point where peaceful co-existence is possible. This in turn, ensures that past investments are protected and that full use can be made of developments in hardware and, particularly in the case of the IBM PC, software.

Appendix

Hardware and software required for reading BBC disks

The floppy disk controller card on the IBM PC and PC XT is based on the NEC 765 controller, equivalent to the Intel 8272. Although this chip is capable of both single-density (FM) and double-density (MFM) operation, only double-density is used on the IBM PC and the controller card lacks the support circuitry for single-density operation. Such circuitry must be added if the controller is to be able to read the standard BBC disk format, Acorn DFS, which uses single-density recording. It should be noted that *any* hardware modification to a system is likely to affect warranty cover or maintenance contract cover and should not be undertaken lightly. Pin 26 (MFM) of the 765 switches to a TTL high level when the device is programmed for double-density operation (the normal state for floppy disk operations on an IBM PC), and to TTL low level when

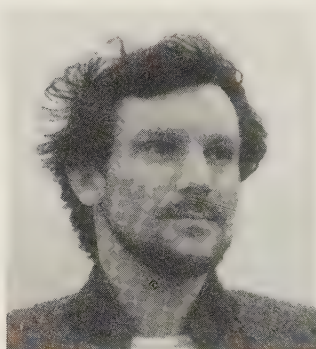
in single-density mode. This signal can be used to switch in (via tri-state buffers) the extra circuitry required for single-density. The minimum required is to switch a divide-by-2 'flip-flop' into the signal path between the phase-locked loop and the 765 (i.e. between U19 pin 7 and U6 pin 22) so that the read window is extended, allowing detection of data pulses recorded in FM format. Once the controller card has been modified to support single-density read operations, software must be provided to read sectors from a BBC disk, interpret the directory structure of the disk and so provide a bridge for data transfer between BBC files and IBM files. Unfortunately, there is no support for single-density operation in the IBM PC ROM BIOS, though it would have been trivial to implement alongside the other disk parameters such as head settle time or sector size, so a machine code routine to read a specified sector must be written as a minimum requirement. The approach adopted in Bristol was to re-write the disk BIOS with support for single-density operation in assembly language (using the IBM Macro Assembler Version 2), then to make calls upon this modified BIOS from a high level language (IBM Pascal Compiler Version 2). The resulting software provides BBC disk directory information, binary and ASCII file transfers, as well as a simple View-to-Wordstar document translation mode. All access to IBM files was simplified by using the Pascal interface, and software development time consequently was kept to a minimum.

LAURIE BURBRIDGE



Laurie Burbridge graduated in electrical engineering at the University of Bristol in 1962. After six years with the Central Electricity Generating Board, working on the design and construction of power stations, he returned to Bristol as a research assistant to investigate novel forms of

rotating electrical machines. This led to the award of a Ph. D. in 1971, followed by an appointment as lecturer in electrical engineering. His interests then shifted from electrical machinery to control systems and the application of computers to real-time control. In 1979 he jointly set up a Microprocessor Applications Group which provided advice to industry on the application of microprocessors to a wide range of industrial problems. His interest in computing continued and in 1981 he joined the University of Bristol Computer Centre as User Support Manager; in 1985 he was appointed Deputy Director of Computing. Laurie has lectured extensively to university colleagues, research students, industrial engineers, managers, and to Ministry of Defence staff.



PAUL BUTTNER

Paul Buttner graduated in biochemistry at the University of Manchester in 1977 and joined the University of Bristol as a researcher in neurobiochemistry. With a 'sideways shift' into computing in 1980, he joined the Department of Computer Science as an electronics technician. In 1984 he took up his present post of Head of Microprocessor Support in the University of Bristol Computer Centre.

11 Moving programs from the Acorn BBC Micro to the IBM PC

A. LOYNS

*Department of Computer Science
The University
Manchester, M13 9PL*

Introduction

This article is concerned with describing the points which must be considered when one attempts to move a program from the Acorn BBC (British Broadcasting Corporation) Micro to an IBM PC.

In order to place the discussion in some context, it is necessary to define exactly what is meant in this article by the terms 'Acorn BBC Micro' and 'IBM PC'. The term 'Acorn BBC Micro' is used to describe the Acorn BBC Micro Model B, fitted with the standard Acorn 8271 Disk Filing System. The term 'IBM PC' is used to describe an IBM PC fitted with the standard IBM Colour Graphics Adapter.

The article does not consider the implications of software used on networked Acorn BBC Micros, although in the majority of cases this networking is hidden from the applications programs and can therefore be ignored.

A comparison of the main features of the hardware configurations is followed by a discussion of file formats and methods of transferring the files from one machine to the other. The article finishes with a comparison of the Basic dialects.

Comparison of the hardware

Disk formats: Acorn BBC Micro

The format adopted by Acorn for their 8271 DFS (Disk Filing

System) was that of a 40-, or 80-track, single-density disk with 10 sectors, each of 256 bytes per track. The disks are single sided—if a double-sided drive is used, the other side is treated as a logically different drive. The Acorn DFS also imposes a limit of 31 files per disk—this being the number of directory entries which fit into the allocated two-sector directory area. Files under DFS are stored contiguously and, therefore, once written are effectively fixed in size. An 80-track disk has a capacity of 200 Kbytes.

Disk formats: IBM PC

The format adopted by IBM for the IBM PC is that of a 40-track, double-sided disk with nine sectors, each of 512 bytes per track. Files are not necessary contiguous on the disk, there being a chain of clusters associated with the directory entry for each file. Each cluster identifies a number of sectors which belong to the file. There is no limit (other than disk capacity and a PC-DOS restriction of 32 Mbytes) to the size of a file. PC-DOS uses a hierarchical file store, there is a limit of 256 files at the highest (or root) directory—but each of these files can be a subdirectory with no restrictions on the number of entries in it. A PC-DOS disk has a capacity of 360 Kbytes.

Screen capabilities

Both the Acorn BBC Micro and the IBM PC use the same CRT Controller chip—the Hitachi 6845. In addition, the Acorn machine uses the SA5050 Teletex character generator chip in one of its display modes.

The Acorn BBC Micro provides eight screen modes, as shown in Table 11.1.

In all but the Teletex mode, the characters are drawn in a 7×7

Table 11.1. Screen modes of the Acorn BBC Micro

Number of colours	Text resolution	Graphics resolution
2	80 × 32	640 × 256
4	40 × 32	320 × 256
16	20 × 32	160 × 256
2	80 × 25	n/a
2	40 × 32	320 × 256
4	20 × 32	160 × 256
2	40 × 25	n/a
16	40 × 25	Teletex graphics

Table 11.2. Screen modes of the IBM PC

Number of colours	Text resolution	Graphics resolution
2	40 × 25	n/a
16	40 × 25	n/a
2	80 × 25	n/a
16	80 × 25	n/a
2	n/a	320 × 200
4	n/a	320 × 200
2	n/a	640 × 200

matrix, positioned within an 8×8 pixel box. There are 95 ASCII characters and 127 user-defined character patterns. In the Teletex mode, each of the 40×25 screen character positions contain an instruction to the SA5050 Teletex chip to draw a character, change colour, draw graphics symbols, or change character height.

The IBM PC provides the screen modes shown in Table 11.2.

In each mode the 256 possible characters are drawn in an 8×8 matrix.

Keyboards

The two machines have different keyboard layouts, but provide similar capabilities. The keyboard on the Acorn machine has a 73-key QWERTY keyboard with four cursor control keys and 10 user-defined function keys; there is no separate numeric keypad. The IBM PC has an 83-key QWERTY keyboard with cursor control keys, function keys, and a numeric keypad; 10 of the keys are user programmable.

Sound capabilities

The sound capabilities of the two machines show the greatest variation; the Acorn BBC Micro has four sound channels, one channel to produce 'noise' and the other three channels to produce tones. Amplitude, pitch, and duration for each channel can easily be specified. Greater control can be achieved by using sound envelopes, which use 14 parameters to denote the rate of change of pitch and amplitude across the duration of the sound. The IBM PC, on the other hand, offers a speaker which can be driven in two ways to produce a note which has a maximum frequency of 595 kHz.

Transfer of software

For any reasonable amount of software, it is not practicable to re-enter the source code on the target machine; apart from being a time-consuming exercise, it is prone to errors. This section deals with methods of transferring the software (both the programs and their associated data files) from the Acorn BBC Micro to the IBM PC.

From the preceeding comparison of the hardware on these two machines, it can be seen that an Acorn DFS disk cannot be read directly on the IBM PC. There are essentially two solutions to this problem:

- (1) use a disk copying service;
- (2) transfer the files using serial communication lines.

A number of university computing centres now offer a service whereby they will take a disk written by one machine and copy the information from that disk onto another disk which is readable on the 'target' machine. There are also a number of service companies offering a similar media conversion service. In order to transfer files using serial communications lines, it is necessary to have a program running in each machine. The most commonly available communications software is called Kermit, which is a file transfer and terminal emulation package designed by Columbia University in the United States. Having decided on the method of file transfer to be used, it is essential to consider the structures of the files on both machines. On the Acorn BBC Micro there are three file structures:

- (1) tokenized files;
- (2) binary files;
- (3) text files.

On the Acorn machine, Basic programs are stored in tokenized files, each Basic keyword (e.g. PRINT, IF, THEN) is replaced by a single byte 'token' in order to (1) increase speed of execution, and (2) to reduce the amount of storage required for the program. As the values of the tokens are unique to the Acorn dialect of Basic, it is necessary to convert tokenized files to text files before transferring them to the IBM PC. This conversion can be done using the *SPOOL command on the Acorn BBC Micro. The structure of a binary file is dependent on the program which reads the binary file, and is therefore independent of the hardware and requires no conversion. The only difference between text files on the two machines is their

interpretation of end of line, the Acorn machine using a carriage return whereas the IBM PC uses the carriage return/line feed pair. The process used for transferring these files (e.g. Kermit, disk copying service) usually takes care of this conversion.

Comparison of Basic dialects

When considering moving a program onto the IBM PC, you should look at the differences in the size and accuracy with which Basic stores its variables on both machines. Acorn Basic uses four bytes to store integers, against the two bytes used on the IBM Basic. With floating point numbers, the range in both dialects is similar, although the accuracy with which the value is stored is different. Acorn Basic offers an accuracy of nine decimal digits; whereas IBM Basic offers a single precision accuracy of six decimal digits and a double precision accuracy of 16 decimal digits. The dialects of Basic also enforce differing rules on how variable names are formed. Apart from the length of the name, the major difference is that IBM Basic does not distinguish between upper and lower case letters in variable names. The appendix to this article consists of three lists which show the differences between the statements in the Basic dialects. The first list shows those statements in Acorn Basic which are not available in IBM Basic; the second list shows those statements in Acorn Basic which are present in IBM Basic but which have slightly differing syntax or side effects. The final list shows those Acorn Basic statements present in IBM Basic but which perform an entirely different function. Statements which are exactly the same in both dialects are not shown. The first list does appear daunting, but it should be pointed out that IBM Basic does provide other constructs, missing from Acorn Basic, which will make the task of moving the program that much easier, for example the `WHILE . . . WEND` construct.

Conclusions

Given the necessary time, a large percentage of programs can be moved from the Acorn BBC Micro to the IBM PC, with varying amounts of ease. The main class of programs which cannot readily be moved are those that take advantage of hardware facilities only available on the Acorn BBC Micro—the Teletex graphics capability, for example. However, by redesigning the program, even this type of

program can be moved. Large, machine-code programs form a class of programs which should not be moved—it will be better to redesign and implement the program on the IBM PC rather than translating any of the code.

Considerable markets for 'add-ons' have been created around both the Acorn BBC Micro and the IBM PC. This article has not dealt with the problems associated with programs which use any of these 'add-ons' with the Acorn BBC Micro; however, it should be noted that some of these 'add-ons' can make the process of moving files from the Acorn machine to the IBM PC much easier (I refer to the myriad of alternative disk filing systems now available for the Acorn BBC Micro; at least one of these systems allows the IBM PC format disks to be read and written).

Finally, in the introduction I defined the Acorn BBC Micro to be the Model B fitted with the 8271 DFS. All the points covered hold equally true for the model B+ and the newer Master series of computers.

Recommended reading and addresses

Acorn BBC Micro User Manual

IBM Basic Reference Manual

IBM PC Technical Reference Manual

Kermit: *Kermit Distribution Service, Lancaster University Computing Service, Lancaster.* The JANET address is SYSKERMIT @ UK.AC.LANCS.VAX1

Appendix

Keywords/statements not in IBM Basic

ACS	EOR
ADVAL	EVAL
ASN	EXT#
BGET#	FALSE
BPUT#	GCOL
CLG	GET\$
COUNT	HIMEM
DEG	INKEY
DIV	LN
ENDPROC	LOCAL
ENVELOPE	LOMEM

MODE	PROC
MOVE	PTR#
ON ERROR GOSUB	RAD
OPENIN	REPEAT
OPENOUT	TOP
OPENUP	TRUE
PAGE	UNTIL
PI	VDU
PLOT	VPOS

Keywords/statements with slightly different syntax

CALL	INKEY\$
CHAIN	POINT
CLEAR	PRINT
CLOSE#	PRINT#
COLO (U) R	RND
DEF FN	SOUND
DIM	STRING\$
DRAW	TIME
EOF (#)	TRACE
IF	USR

Keyword/statements which are totally different

GET

AIDAN LOYNS



Aidan Loyns graduated from the Department of Computation at the University of Manchester's Institute of Science & Technology in 1976 and started work at the University of Manchester's Department of Computer Science in the computer-aided computer-design research group, producing software to be used in the design and manufacture of digital systems. In

1983, he joined the Barclays Microprocessor Unit at the University of Manchester, to work on all aspect of microcomputing, with special reference to the needs of scientific word processing and the use of laser printers in an office environment.

12 Economic literacy and strategic planning for the executive—the role of the IBM PC

K. G. LUMSDEN and A. SCOTT

*The Esmee Fairbairn Research Centre
Heriot-Watt University
Chambers St
Edinburgh, EH1 1HX*

Introduction

‘Discuss the problems governments face using monetary and fiscal policy in running their economies to achieve desirable societal goals.’

‘You have just assumed control of a liner company in today’s depressed world shipping market. Your goal is to maximize the net worth of this company over a ten-year period. What is your strategic plan?’

Neither of these questions is easy to answer, nor are they trivial subject matters to teach, but they are the types of topics which are prominent in business executive courses. Because of their complexity, however, they are not readily amenable to conventional teaching techniques. The computer has played a vital role in filling this pedagogical gap, and the advent of the microcomputer has lowered the cost and increased the convenience of the computer as a teaching tool.

There are two areas where the microcomputer has a comparative advantage in teaching:

- (1) simulating real life situations; and
- (2) providing the means of analysing data.

In the field of economics we have developed both macro- and micro simulations, and set up a database of national income and related statistics for over 20 countries since 1959, with a set of utility programs which permit immediate analysis of the data. The wide adoption of the IBM PC has greatly assisted in the dissemination of the programs.

Simulations

Two characteristics of the simulation approach contribute to its effectiveness. First, the results of policies are instantly available in the computer output; this provides an immediacy of feedback which is missing from conventional teaching techniques. Second since the simulations are normally run by teams consisting of four or five people, they provide a unique platform for discussion and exchange of ideas. Many personnel managers and instructors have observed levels of interaction among managers running the simulations, which they had never previously experienced. At the macro level, the simulations are able to deal with the dynamic interdependencies of the macroeconomic system. At the micro level, the simulations can capture the complex nature of business decision-making, including prediction, investment decisions, pricing policies, the selection of new markets and products, and the deployment of budgets. This approach has the potential for teaching the type of lessons which managers can typically only obtain from hard experience, and often only from making costly mistakes in real life.

Macrosimulations

TEFRC (The Esmee Fairbairn Research Centre) has developed a series of macrosimulation computer models (Lumsden and Scott 1981, 1982, 1984, 1987; Lumsden *et al.* 1983). The overall objective of the macrosimulations is to help students to learn basic macroeconomics in an interesting fashion. They are not, however, simple substitutes for the written or spoken word, as they have dimensions which no textbook or conventional classroom delivery can replicate. The computer continuously solves a set of non-trivial simultaneous equations when prompted by values of specified variables—the policy variables—determined by the team of players. The output is multi-dimensional; it tells what happened to all the important variables in the system as a result of these policy changes. Thus,

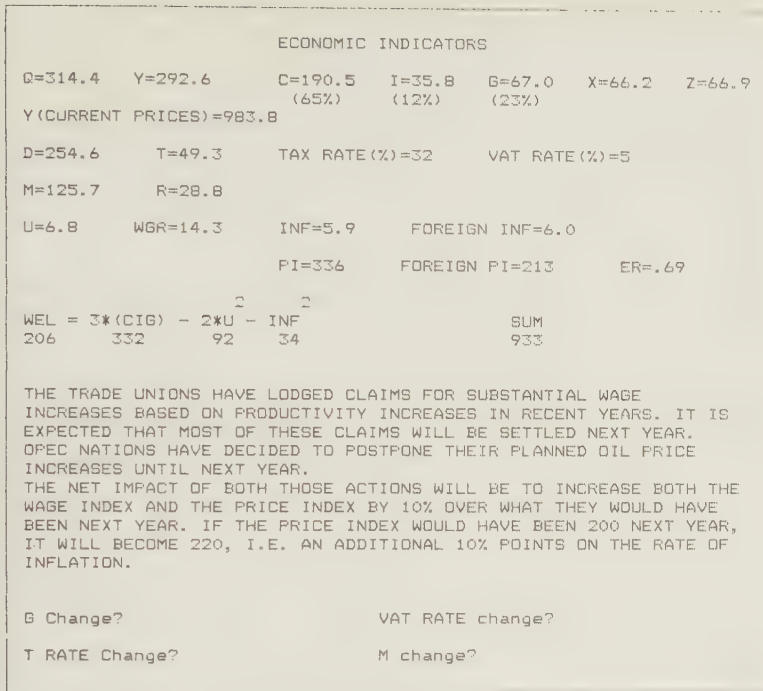


Fig. 12.1 A screen used in Running the British Economy

playing the simulation provides students with a feeling for the dynamics of a macroeconomic system; they are able to experience the problems faced by real-world policy makers and are forced to put their understanding of macroeconomics to work in trying to solve these problems. An example of one of the screens produced in such a simulation is shown in Fig. 12.1. In conventional teaching, students too often end up with a static view of macroeconomics, which results in a lack of comprehension of many of the most important and interesting macroeconomic issues.

Microsimulations

Shipping Management Model—Stratship

Stratship is a program which simulates the process and consequences of higher management decision-making in a shipping company. The

purpose is to teach modern management skills by integrating accounting, economics, finance, marketing, quantitative methods, and strategic planning, into a realistic simulation.

Typically, the model is run in management development seminars by teams of 4–6 executives who take over the running of a shipping company at a crucial stage in its development. They are provided with a short history and route accounts of the company. The microeconomic decisions that have to be made are linked to a macroeconomic world situation by way of business cycles. The team runs the company for a certain number of quarters, in each of which decisions have to be made, based on the company's past results and on information regarding market share and capacity utilization. Background information is available in the form of a 'brokerage service', which provides information on vessel prices and new building construction lag, charter rates, competing liner rates, sea cargo trade indices, interest rates, and fuel prices.

Decisions on the fleet structure include ordering new vessels, buying vessels on the spot market, chartering vessels in, selling vessels, chartering vessels out, and scrapping vessels. Operational decisions include specifying new routes, or modifying existing routes, allocating vessels to routes, determining marketing expenditure, and setting rates for carrying cargo.

The structure of the simulation and the decision-making process was derived from a series of interviews with leading company executives in the shipping industry throughout the world; early versions of the model were tested on groups of managers in the shipping industry, and the model refined in light of experience.

Because of the general nature of the management problems which it presents, Stratship is now widely used in general management programmes in which participants have no experience of the shipping business.

Analysing real-life data—Macroeconomic Database

The Macroeconomic Database (Lumsden *et al.* 1983) permits users to handle real-world numbers and analyse national income data by a series of in-built utility programs. The raw data consists of national income and related statistics (national income, consumption, investment, government, exports, imports, inventories, national income at constant prices, consumer price index, index of

production, index of wages, exchange rate, money supply, rate of interest, population, employment, tax revenue, government expenditure, unemployment) for over 20 countries for the period 1959–85. This data is input in such a fashion that the series can easily be updated each year as new data emerges. A great deal of effort has been devoted to ensuring that the data series are complete and consistent, thus avoiding the tedium and problems researchers experience in collecting data from diverse sources. The development of the Macroeconomic Database is a response to the demand for more analysis and more real world examples in economics teaching.

The utility programs allow users to analyse the raw data by:

- (1) creating new variables by adding, subtracting, dividing, or multiplying, existing data;
- (2) creating new variables by taking first differences of existing data; and
- (3) running linear regressions on two variables, with or without time lags.

The Database therefore permits users to test various hypotheses and relationships using real data; for example, the relationship between inflation and unemployment, or the impact of changes in national income on consumer expenditure for a large number of countries.

Concluding remarks

Measuring the efficacy of innovative teaching techniques in a rigorous fashion is complex and difficult, and no large-scale experiment to evaluate the simulations has been conducted. However, the market response to the simulations is an indication of their effectiveness. Over 5000 students studying A Level Economics take part in the annual competition 'Running the British Economy' each year; over 2000 macroeconomic simulation computer packages have been sold to date.

Thousands of business executives have been subjected to the macroeconomic simulations in executive programmes throughout the world. In the UK they form an important element in the development programmes of such companies and institutions as ACL, BP, Barclays Bank, Civil Service, HP, IBM, Land Rover, Marks & Spencer, National Westminster Bank, Reckitt & Colman,

Rolls Royce, and United Dairies. Stratship is used in shipping programmes from Hong Kong to New York; it also features in strategic planning courses for non-shipping executives in some of the world's leading business schools. In subjective evaluations of business executive programmes throughout the world, both the macro- and microsimulations have received top ratings as pedagogies.

References

- Lumsden, K.G. and Scott, A. (1981). *Basic Macroeconomic Models*. Longman, London.
- Lumsden, K.G. and Scott, A. (1982, 1983, 1984, 1987). *Running the British economy*. Longman, London.
- Lumsden, K.G. and Scott, A. (1984). *Macroeconomic Database*. Longman, London.
- Lumsden, K.G. and Scott, A. (1986). *Running the US Economy*. Heriot-Watt University, Edinburgh.
- Lumsden, K.G., Scott, A., Edwards, R., and Helgesen, K. (1983). *Stratship*. Heriot-Watt University, Edinburgh.



KEITH LUMSDEN

Keith Lumsden is Professorial Fellow and Director of The Esmee Fairbairn Research Centre (TEFRC), Heriot-Watt University, Edinburgh. He is also Affiliate Professor of Economics at the Institut Européen d' Administration des Affaires (INSEAD) in France, and Professor of Economics at Stanford University's Advanced Management College in California. He taught in Stanford Economics Department and Graduate School of Business for 17 years before returning to Edinburgh in 1975 to devote most of his time to research in TEFRC. The Centre's primary research is concerned with the development and determination of the efficacy of innovative teaching techniques designed for school pupils, university students, and business managers and executives.

He has published 14 books and over 60 articles in professional journals, and has been jointly responsible for the development of both micro- and macroeconomic models for use on microcomputers.



ALEX SCOTT

Alex Scott is Senior Research Fellow at The Esmee Fairbairn Research Centre. He has been closely involved in the development of simulations of both the economy and business firms for many years. His academic research interests are currently in the economics of education, energy economics, and the economics of government and industry; he has published widely in these fields.

13 An array processor for the IBM PC

R. E. BURGER

*Cambridge University Engineering Department
Trumpington Street
Cambridge, CB2 1PZ*

Introduction

This article describes a new computer peripheral for scientific users of the IBM PC XT/AT. A single board IBM PC array processor, the AP1000, has been developed for applications requiring floating point processing on large data arrays. Expedient use of the AP1000 dramatically improves the computational throughput of an IBM PC.

Firstly, the need for a fast arithmetic coprocessor for the IBM PC is illustrated with specific reference to the field of digital image processing. This is followed by a brief description of the array processor architecture, emphasizing application-oriented considerations for its efficient use. Lastly, we look at software support for the array processor and examine the price/performance advantages of an IBM PC-based scientific workstation.

Image processing on the IBM PC

Digital image processing has traditionally been confined to implementation on mini- and mainframe computers, or relatively expensive special-purpose systems. This has been due to the demanding computational speed and storage requirements placed on systems capable of executing image processing algorithms. The rapid growth of digital image processing over the past 20 years has been made possible by the ever improving price/performance ratio of hardware capable of rapidly processing the large amounts of data contained in a single medium-resolution image. The recent

availability of inexpensive and reliable 16 and 32 bit microprocessor-based systems has projected image processing into the realm of personal computers.

The advantages inherent in designing an image processing system around a microprocessor-based system include the relatively low cost of the associated hardware (system unit, array processor, and framestore—a large array of video memory for storing images in digital form) and support software. Moreover, the flexibility of such systems in terms of the range of available software, enables them to perform tasks other than image processing, making them an attractive option for both academic and industrial environments. The IBM PC XT, and more recently the IBM PC AT, have become market leaders in the field of microprocessor-based computing for scientific applications and are, therefore, obvious candidates for development as workstations for the processing and presentation of digital image data.

Whereas the combination of 640 Kbyte directly-addressable memory, coupled with an IBM Winchester disk, makes the IBM PC suitable for many scientific applications, the speed at which it is capable of performing floating point calculations precludes its use for many DSP (Digital Signal Processing) applications requiring fast CPU throughput. The floating point hardware currently provided as an option for the IBM PC family consists of the Intel 8087/80287 (XT/AT) 'maths coprocessors'. This hardware enhancement, while providing a substantial processing speed improvement over the arithmetic capabilities of the host processors (8088/80286), is still a little slow when the IBM PC is used for interactive image processing.

A commonly-used algorithm in the field of digital signal processing is the Fast Fourier Transform (FFT). Benchmark timings for this algorithm were run on various systems and are illustrated in Table 13.1. The times indicated are for a single iteration of a fairly efficient Fortran coded 1024 point complex FFT (Cooley and Tukey 1965).

The extremely good accuracy of the IBM PC is a result of the 80-bit internal floating-point representation of the 8087/80287 coprocessors. It can be seen from this benchmark comparison that the CPU throughput of the IBM PC AT is a factor of 20 slower than the VAX 11/780 and 50 times slower than the IBM 370/165.

Although waiting a mere four seconds for the above computation may seem entirely acceptable, consider the implications for

Table 13.1. Benchmark timings for the FFT algorithm (courtesy of W.O. Saxton, Cambridge University Department of High Resolution Electron Microscopy)

Machine/Config.	Time (ms)	Accuracy (p.p.m.)
IBM 370/165 G1	87	412
" " HExt	71	412
VAX 11/780*	199	14
IBM PC AT*	3880	10

*indicates the presence of a hardware floating-point accelerator.

two-dimensional transforms. Based on a direct* extrapolation, $N \cdot \log(N)$, from the figures in Table 13.1, the IBM PC AT + 80287 requires approximately 30 minutes (excluding disk I/O overhead!) to compute a two-dimensional complex FFT on a 512×512 digital image array. Extensive effort at assembly level coding of the FFT algorithm has failed to reduce this time by more than a factor of two.

Although there are many objective criteria for calculating image fidelity, interactive image enhancement and restoration typically requires multiple iterations of a filtering operation in the spatial frequency domain until the subjective 'best' image is produced. For the IBM PC to be successfully integrated into an image-processing workstation, the results of filtering operations must be available for viewing at least an order of magnitude faster than the times indicated above. This consideration provided the motivation for the array processor design.

Array processor architecture

For the purposes of this article, the term 'Array Processor' (AP) refers to a single dedicated peripheral processor board, interfaced to the host, and designed primarily for high-speed 'number crunching'. The 'array' refers to the one- or two-dimensional data processed by the AP; this is quite distinct from the concept of an array processor in which the 'array' refers to a matrix of distinct processor elements, all operating in parallel using SIMD or MIMD processors (Single/Multiple Instruction control of Multiple Data streams: two types of parallel array processor architecture). A block diagram (Fig. 13.1) of the AP1000 array processor for the IBM PC, appears in this article.

ARRAY PROCESSOR ARCHITECTURE

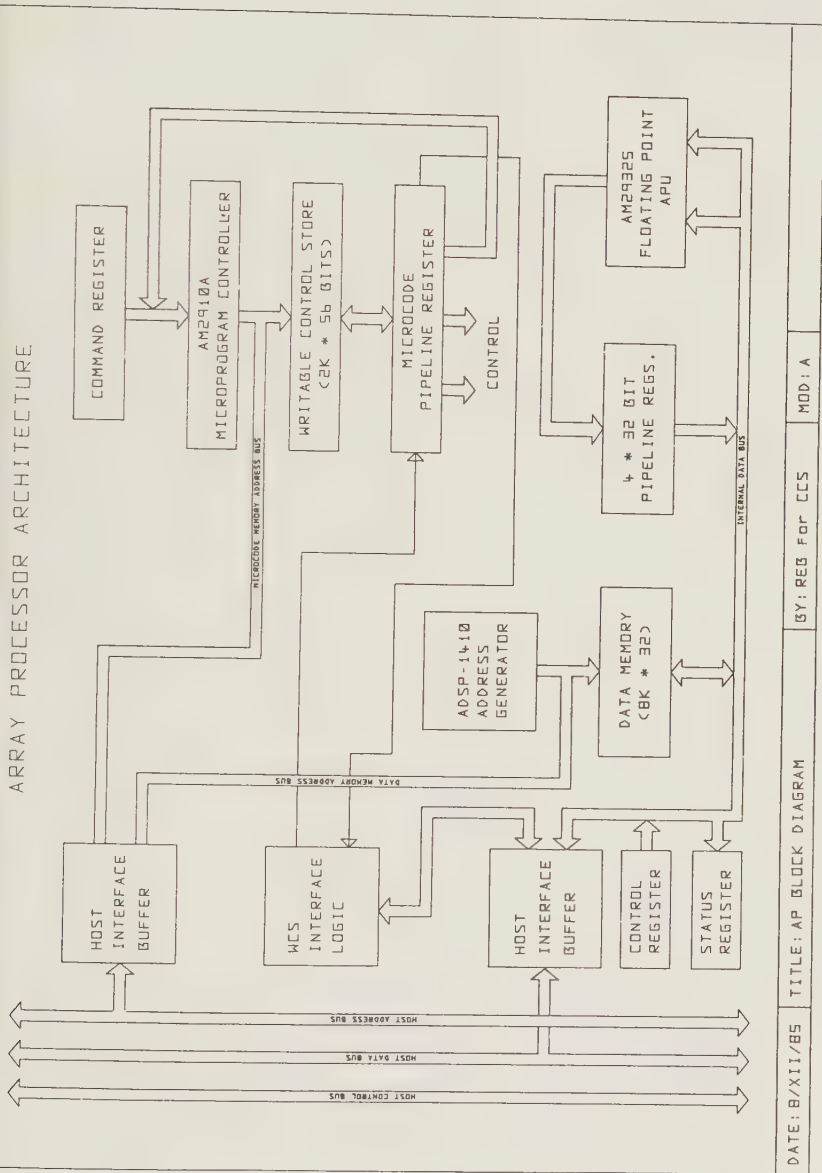


Fig. 13.1
Block diagram
of the AP1000
for the IBM
PC.

Main Features

- APU (Arithmetic Processing Unit) supports full 32 bit IEEE 754 data format
- Single plug-in board compatible with IBM PC XT/AT
- 150 ns cycle time
- Memory-mapped architecture
- 8k by 32 bits of data memory
- 2k by 56 bits of writable control store
- Fortran and C callable routines for DSP and vector operations
- User programmable with optional microcode assembler

Processing speed

The most important single factor influencing array processor performance (computational throughput) is technology. Traditional 'bit-slice' solutions (a hardware architecture in which the basic component integrated circuits operate on sub units, or slices, of the whole word length) are increasingly giving way to single chip VLSI multiplier/accumulators (MACs) and integrated DSP engines. It is the recent advent of such silicon technology that has made it possible to design a high-performance floating point array processor onto a single IBM PC expansion board. The AP1000 features two VLSI DSP-orientated integrated circuits. The arithmetic processing unit on the array processor is a single chip ECL (Emitter Coupled Logic) floating point MAC from Advanced Micro Devices (AMD), the AM29325. This integrated circuit is capable of executing a full 32-bit floating point operation (add, subtract, multiply) every 150 ns. Compare this to the IBM PC AT Intel 80287 (approximately 16 microseconds at 6 MHz for a floating point multiply [*Intel Microsystem Components Handbook*, Vol. 1, 1985, p. 4.58] and one can begin to appreciate the computational advantage of this hardware! Address generation for the 8K words (32 Kbytes) of on-board data memory is handled by a CMOS Address Generator from Analog Devices, the ADSP-1410.

Realizable array processor throughput (measured in real elapsed time) is a function of both the actual AP computational speed (usually measured in megaflops—millions of floating point operations per second) and the time required to bus the data into and out of the AP memory. The IBM PC AT external bus supports a maximum data transfer rate to/from the AP1000 of about 1 Mbyte per sec. To see the significance of this we will consider the

time taken to execute a 1024-point complex FFT. For this operation, 8096 bytes (4 bytes real, 4 bytes imaginary, per data point) must be loaded and then subsequently unloaded to/from the AP1000. Thus, at 1 Mbyte per sec this represents approximately 16 ms data transfer overhead. The 1024-point complex FFT takes only 13 ms to execute on the AP1000 for a total real elapsed time of 29 ms. Data transfer often represents a significant percentage of the execution time, a fact often hidden behind impressive megaflop specifications.

Accuracy

The desired accuracy of an array processor is an issue of number representation. Traditionally, there have been three types to choose from—fixed point, block floating point, and full floating point. Fixed point systems are generally faster and cheaper than floating point, since the hardware requirements are considerably less demanding. Block floating point solutions have, in the past, represented a good compromise between fixed point integer arithmetic and full floating point solutions. The availability of fast VLSI MACs conforming to the IEEE P754 standard for floating point numbers will make block floating point architectures for DSP and vector processing virtually obsolete.

Real-world data is often only digitized to 8-bits precision and it is thus reasonable to ask whether such applications really require the accuracy and dynamic range available with floating point arithmetic. Some trivial first-order DSP operations can, indeed, be performed using integer arithmetic. However, the majority of iterative second- and third-order operations (the FFT is a familiar example) mandate the use of floating point numbers. Matrix operations aimed at solving systems of linear equations may require accumulation to 64-bit floating point precision.

Software support

Microcode library

A library of over 100 microcode routines, callable from either Fortran or C, is available for the AP1000. These routines include vector and matrix operations (real and complex), and support for many DSP and digital image-processing algorithms. Use of the array processor is designed to be as transparent as possible to the programmer; a trivial Fortran-coded example to multiply two

vectors is illustrated below. It illustrates the ease with which existing vectorizable code could be modified to take advantage of the array processor.

PROGRAM EXAMPLE

```
C
      REAL X(1024, Y(1024)
C Initialize the array processor library
      CALL APINIT
C
C The user codes up the input or initialization of data arrays
  here—
C For example
      READ (1) X
      READ (2) Y
C
C Multiply arrays together and store result in Y
      CALL RMUL (X,Y,Y,1024)
C
C Write results back to disk
      WRITE (3) Y
C
C Release the array processor
      CALL APEXIT
C
      STOP
      END
```

Microcode assembler

Although the standard microcode libraries will be sufficient for most applications, enterprising programmers may decide that their application justifies a custom microcode routine. To accommodate this, a high-level microcode assembler and loader are available, enabling the programmer to download custom microcode into the AP's writable control store. In addition, new routines can be thoroughly tested and debugged using a software board-level simulator.

Price/performance considerations

Benchmarks

The first section of this article provided an example of the time taken

to execute a two-dimensional FFT on the IBM PC AT as motivation for designing the AP1000. The following timings have been obtained on the AP1000 for a 512×512 real FFT:

Computation time:	4 secs
Host/AP data transfer time:	5 secs
Array transpose time:	5 secs
Total:	14 secs

This represents a real-time speed improvement of greater than two orders of magnitude over the Fortran-coded example. Note that certain assumptions have been made here consistent with the Fortran example given previously. Most significantly, no allowance has been made for disk transfer time (56 secs on an IBM PC AT); this is possible on the IBM PC AT although not on the IBM PC XT, using the extended memory facility of the Intel 80286.

The following chart (Table 13.2) is adapted from an article that appeared recently in the Proceedings of the IEEE. It makes interesting reading but should probably not be taken too seriously!

The AP1000 array processor for the IBM PC was designed and developed by Camtrel Computer Systems Ltd, Cambridge, England. Further enquiries concerning availability of the AP1000 should be addressed to Data Translation UK Ltd (0734-793838) who are currently marketing this product.

Table 13.2. Benchmark comparisons between the IBM PC AT (with and without AP1000), and other computers (after South and Dolecek 1985)

Computer	(k\$)	(Mflop)	(M\$/Mflop)
IBM PC AT	4	0.00037	10.8
IBM PC AT + 80287	4.5	0.0073	0.62
IBM PC AT + AP1000	10	5.0	0.002
VAX 11/750	80	0.057	1.4
DEC KL-10	480	0.18	2.7
CDC-7600	4500	4.6	0.98
Cray-1S	11 000	18	0.61
Cray XMP	15 000	53	0.28

References

- Cooley, J.W. and Tukey, J.W. (1965). An algorithm for the machine calculations of complex fourier series. *Mathematical Computation* **19**, 297-301.
- South, H.M. and Dolecek, Q.E. (1985). Signal processing and engineering analysis on personal computers. *Proceedings of the IEEE* **73**, (12), 1826-37.

Note

Readers should remember that this is one of a number of array processors available for the IBM PC. Each one has its own characteristics . . . Editor



RUDY BURGER

Rudy Burger received his primary and secondary school education in England. He subsequently attended Yale University, Connecticut, USA, as an undergraduate between 1976 and 1980. During 1978 Rudy obtained an academic leave from Yale and joined a research group at North American Philips Corporation investigating solid state laser technology for the Philips video disc system. Rudy graduated from Yale with a B.Sc. in

electrical engineering in 1980 and was elected to the Sigma-Xi scientific honorary society.

He worked at IBM Research Laboratories in New York from 1980 to 1982 on the design and fabrication of VLSI bipolar circuits and devices, and was awarded an IBM Research Divison Award in July 1982 for his contribution to the project. In 1982 he returned to Yale for a one-year M.Sc. programme in electrical engineering, on a full scholarship from the Sheffield Foundation. He received an M.Sc. in May 1983.

In 1983 he returned to the UK to attend a three-year Ph. D. programme in the Cambridge University Engineering department. He is currently completing his thesis on ultrasonic image processing and is an elected Bye-Fellow of Magdalene College, Oxford.

He founded Camtrel Computer Systems Ltd (CCS) in 1985. CCS is a Cambridge-based research and development company which designs and develops image processing hardware and software for microcomputers.

14 Computing and preventive cardiology with an IBM PC

R. W. JONES

*St Mary's Coronary Flow Trust
Waller Department of Cardiology
St Mary's Hospital
Praed St
London, W2 1NY*

Introduction

Uncertainty about the clinical value of computing in medicine persists in a profession where human contact and understanding are inseparable from successful diagnosis and treatment. Some commentators believe the drift toward technology threatens established clinical care, and that the techniques used to discover and manage disease dominate the clinical response rather than the needs of the sufferer. Computing is associated in many peoples' minds with an approach to human affairs that lacks compassion. The computing *aficionado* believes such a view to be at best ill-informed, and at worst Luddite. It is likely that computing has not yet come of age as far as health care is concerned, but it does promise to provide the means by which new demands on health care, such as effective prevention, can be met. The appearance of portable and powerful personal computers, like the IBM PC, makes it more likely that computing will lead to better health care.

The problem

The Alma-Ata Declaration on health care, by the World Health Organization in 1978, described the relative distribution of health care in an industrialized society: 5 per cent of health problems were dealt with by specialist hospital-based medicine; 20 per cent by primary medicine, the general practitioner; and 75 per cent of

problems were coped with by the sufferers and their families. Now, in hospital medicine computers have been applied readily to the measuring, interpretation, and storage of physiological parameters, as well as the organization of patient details. As a result, the application of new technologies is motivated by the need to respond successfully to those acute and life-threatening illnesses that make up much of the 5 per cent of health problems cared for in hospital. However, despite increasing investment, there is no convincing proof that hospital care leads to profound changes in the level of health care in the general population. Moreover, the belief that hospital care could change the level of general health may come from an understandable admiration for the hospital service rather than from an understanding of how disease tends to develop amongst us. Whether a specialist service is affordable without economic restraint is a separate question, but should be the stimulus to finding the way in which those 95 per cent of health problems outside hospital could be better managed and hospital admission avoided. From the medical standpoint, this stratagem is dependent upon the belief that amongst those 95 per cent of problems cared for by primary-care medicine and the family, there are the antecedent illnesses and risks that lead to dangerous illness. Can screening and early intervention deflect a disease process and prevent serious, and possibly irreversible, damage to the health of an individual?

Consider coronary artery disease that leads to angina and heart attack. Illness due to disease of the circulation is the cause of half of the deaths in our society, and of these deaths half are due to coronary artery disease. It is a disease that can kill people suddenly: one-third of those cases that present for the first time do so as a death. Of these sudden deaths, 70 per cent occur at home and a handful reach hospital, while of those that do survive the first appearance of the disease, life expectation remains poor despite considerable advances in diagnosis and treatment.

General practitioners can expect 47 per cent of their patients suffering from coronary artery disease to die within five years. It has also been found that general practitioners can be unaware of half of those patients on their lists with the disease, so the likelihood of knowing about those at risk is less. If it caused death at or near the end of our expected life span, the impact of this disease on our society would be lighter, but unhappily it has a predilection for males of a relatively young age, and who can still have dependents of school age.

The development of the disease is slow and without symptoms. The eventual clinical appearance is dramatic and life threatening. There are recognizable risks associated with both the development and the appearance of the disease that could be influenced, but irrespective of the time and money spent on the treatment of overt coronary artery disease little will be gained until prevention can be made effective. Primary-care medicine, and the families it cares for, need a system that is cheap, approachable, responsive, and intelligent, to support preventive medicine.

Questions and answers

How can coronary artery disease be overcome?

Human and environmental factors can be recognized: sex; age; excessive weight; raised blood pressure; raised blood fats; smoking; family history; and personality.

How might these factors become recognized?

If you believe that diet alone is the root cause of all the risk then it would be pointless to seek risk, as a change in the sociopolitical climate leading to changes in dietary habit would suffice. If diet is not the root cause, or waiting for a change in eating habits is unacceptable, then the general population can be screened and appropriate advice or treatment offered to an individual found to be at risk.

Once risk is found, can intervention help?

Many believe this to be the case but there is no proof, and statistical confirmation may never be possible in a society whose collective habits change faster than the evolution of coronary artery disease.

Is intervention safe?

Changes in lifestyle should be less risky than medical treatment. Nevertheless, ecologists often describe unexpected disasters when habitat or behaviour are changed by external influence, and there is no reason to believe that we are free from such risk. Whatever the intervention used, safety or, conversely, danger can only be given a value when compared with the gain for an individual and for society.

What is needed

Better health care needs to be promoted throughout the health service in a way that reflects the relative distribution of illness and risk of illness.

Accepting that prevention is the best option, a system of health assessment and education must be developed to work in peoples' homes and work places, and which functions in harmony with general practice. To enhance prevention there has to be a new approach to health care which is less dependent upon the need of an illness as a passport to health assessment, and avoids excessive use of expensive resources such as doctors and nurses. It should also be responsive to the 75 per cent of health problems coped with by the individual and family. We need to know if recognition of health problems and subsequent intervention can affect health: the system has to be monitored continuously and provide health service administration with such data that will aid the development, delivery, and support of health promotion and disease prevention.

Computer-based management systems can provide the tools for the task.

The choice of database is of obvious importance. MUMPS, the programming language, and COSTAR, a primary-care database, were both developed in response to medical needs. MUMPS will run on the IBM PC AT and is being adopted by medical administration in the United Kingdom. Public Domain COSTAR is a system of 1500 programs written in MUMPS with a directory of at least 8000 medical terms. Running on the IBM PC AT it becomes affordable and is capable of communicating with medical administration. Medical Query Language is a facility for the interrogation of the database, and with it COSTAR can produce a report containing demographic variables, present health status and health problems, environmental and other risk factors, for both an individual and a community. It has a comprehensive series of statistical procedures making the epidemiological study of the data feasible. Once implemented, a system of this power can direct intervention and education.

Collection of data fails when either the effort needed is too great relative to the expected reward, or the process has a deleterious effect upon those who are screened. When compared with the number of people to be screened there are few who will be found to be ill or at risk. Doctors are trained to recognize overt disease and treat it, and

can be expected to lose motivation if they are unable to show their skills. Driven by their desire to help, doctors may create disease where none exists. Moreover, people associate illness with doctors and adopt the role of patient when coming into contact with them, which can lead to the danger of causing iatrogenic, or medically-induced, disease.

How can disease be sought without doctors being involved directly and acting as a 'dangerous drug'?

An answer would be to mimic their skills in history taking, by the use of logic programming. The decanting of medical expertise from the clinical environment into a computer system allows both widespread application and the avoidance of many of the problems described above. Questionnaires have been developed to mimic medical history taking but in the main they have been used in research, and have not been accepted for usual clinical care because of the considerable work that the management of written questionnaires creates. A shift of written to computerized questionnaire does not answer this problem completely, as the expertise upon which the questionnaire is developed has to be applied to the answers at a later stage, and this separation reduces its ability to respond to changes in clinical opinion. Decision trees combine the two but are restricted by biasing where the number of variables is large, as is found in multifactorial coronary artery disease. Biasing can be reduced where validation of the sub-groups is used to weight the decision-tree nodes statistically. So, despite their weaknesses, trees can help make effective clinical decisions, but they do not represent the heuristic nature of clinical knowledge well.

The algorithmic languages, Cobol, Fortran, Pascal, and ADA are particular and unyielding in the way they organize the function of the machine; not unlike the way in which clinical decision trees produce an answer. Recent developments in programming have led to high-level declarative programming languages that need less program detail, and use descriptive statements similar to those that doctors are likely to make in reaching their diagnoses. By containing expert knowledge as part of the program structure, the diagnostic procedure is copied and, like any diagnostic method, is capable of review in the light of new knowledge. Micro-PROLOG is the logic programming language with which our group are writing 'intelligent' questionnaires. Dr Peter Hammond and Marek Sergot at Imperial College, University of London, have expanded Micro-PROLOG to form APES, designed to ease the building of expert

systems. APES runs on both the IBM PC XT and IBM PC AT. Programs compute a deduction from information presented in a number of logical statements. This deduction or inference can be validated by experience and adjusted to fit, and, from an expanding database, rules can be made and inferences drawn that do not reflect accepted clinical opinion and which, with validation, may lead to original views upon the causes of the disease. The making of knowledge rules can be shown in the following manner:

Risk factor for heart disease is obesity,
Risk factor for heart disease is family history of heart disease,
Blood fats analysis is investigation for heart disease.

If risk factor for Disease is First risk factor
and risk factor for Disease is Second risk factor
then Disease is likely.

If Disease is likely,
and Investigation of Disease is Investigation
then do Investigation.

A series of such inferences can be produced from the data gleaned by a questionnaire, and can lead to an overall description of an individual's present and future cardiac health. At present our questionnaires are literate and their use made easier by a purpose-built keyboard with four keys, each with an LCD alphanumeric screen for prompting, and a numeric pad. Questions are then displayed on the screen (see Fig. 14.1).

The interpretation of written questions remains taxing for us all and interview-assisted questionnaires have proved more reliable, but the need for an interviewer, to improve the sensitivity and specificity of a questionnaire, undermines the usefulness of the technique. The case for a purely literate questionnaire, whether in paper or computer form, is further weakened when you consider who in our society is most likely to suffer from the disease, and for whom the questionnaire system should be designed. The number of deaths from coronary heart disease remains highest amongst unskilled workers represented by the income definitions of social classes C and D. Having little use for the nuances of grammar and sentence construction, they prefer the speed and immediacy of the graphical presentation of information. The making of questionnaires in video would present symptoms and related queries in a graphical form, and avoid both boredom and disinterest on the part of the interviewee, as well as reducing the misinterpretation of questions.


```

+-header-----+
|ST. MARY'S AND IMPERIAL COLLEGE QUESTIONNAIRE PROJECT|
+-current question-----+
|Under which circumstances do you get the chest pain|
+-painmenu-----+
|walking uphill or in a hurry|
|walking at an ordinary pace|
|standing still or sitting|
|end|
|enough|
+-current question-----+
|What do you do when you get the pain|
+-either-----+
|carry on walking|
|stop or slow down|

```

Fig. 14.1 Screen display of part of a questionnaire.

The physiological parameters associated with coronary artery disease include: blood pressure; blood fats and glucose; and particular variables in the electrocardiogram recorded at rest and on exercise. The first can be measured from the finger or arm without medical supervision while, with new paper strip and

optical technologies, blood fats and glucose can be measured rapidly, providing an immediate result to be sent to the database directly.

Computerized electrocardiography is established, and interpretive programs are available. The impact of portable computers upon signal processing has been considerable in recent time, and has brought the accurate measurement of physiological parameters, such as electrocardiographic variables, from the laboratory to the 'bedside'. Specific changes in the electrocardiogram at rest and during exercise are found to be powerful predictors of later cardiovascular events, and of particular interest are the abnormal exercise electrocardiograms found in asymptomatic men. This group provides an exciting opportunity to study the effect of clinical intervention on long-term cardiac health. The ethical difficulty of treating asymptomatic people with disease is eased by the results of large epidemiological studies (Oslo and MRFIT) which showed improved cardiac mortality among men at high risk who were treated medically. A number of years have been spent building a portable, cheap, accurate, and convenient exercise electrocardiograph suitable for epidemiological studies. The design by Dr Malcolm Clarke of Imperial College, University of London, uses a Z80-based processor running in tandem with a personal computer supporting a 10 Mbyte hard disk. Twelve channels of unipolar data are recorded with a Wilson Central Terminal as a reference. The electrocardiograms are digitized at 250 Hz with 11 bit accuracy while the signals are multiplexed prior to digitization and transferred to the personal computer by interrupt routines. Recordings (of duration 30 s) are made each minute throughout a standard protocol of exercise, as this length of recording gives a sufficient number of signals to improve the signal to noise ratio by averaging. During recording the exercise procedure is controlled by the personal computer which manages the Z80-based processor, the treadmill protocol, the output of a three-channel pen recorder, and the input of clinical symptoms and blood pressure recordings. A channel of data is displayed on the screen along with the timings of clinical symptoms and blood pressure recordings.

The first and second differentials of the averaged signals are derived, to aid fiducial point recognition. Once found for each channel at each stage of exercise stress, the changes in the electrocardiogram variables can be plotted against time, changes in pulse rate or blood pressure, and the work done by the subject. The

diagnostic power of the test is enhanced further by including the recorded onset of symptoms.

In the general population, where relatively little disease is expected, the specificity of a test has to be high to avoid false positive results. By combining a series of tests, including the questionnaire, the accuracy of the complete system is expected to be greater than any individual part. When compared with analogue recorders and measurements by graticule and lens, computerized exercise electrocardiographs are capable of significantly greater accuracy. We have found that a sufficiently sophisticated exercise electrocardiograph can be built utilizing a personal computer.

Conclusion

The data gained from the questionnaire, and from the examination of the blood and heart, is entered into the COSTAR database. Interrogation of the system reveals those with overt disease as well as those who could develop the disease in the future. The setting of the system in primary care allows the general practitioner to organize intervention and be aware of the outcome. By communicating with secondary health care and all levels of the health administration, the system allows the practitioner to intervene with greater effect and without professional isolation. Also, in response to the data collected, health services can direct resources that are appropriate to the health needs of a local community. To introduce the system as part of the stratagem for the prevention of ischaemic heart disease will be a large task, but the rewards, in health terms, will be considerable.

References

- (MRFIT) Multiple Risk Factor Intervention Trial Research Group (1982). Multiple risk factor intervention trial. Risk factor changes and mortality results. *JAMA*, **248**, 1465-77.
- (Oslo) Hjerman, I., Holme, I., Byre, K. V., *et al.* (1982). Effect of diet and smoking intervention on the incidence of coronary heart disease. *Lancet*, **2**, 1303-10.

RUSSELL JONES

Russell Jones received his M.B.B.Ch. from the University of Wales in 1972, and became house physician to the Medical Unit at the University Hospital of Wales. This was followed a year later by the post of House Surgeon in

Urology at the same hospital. In 1973 Russell moved to London where he was appointed Senior House Officer in General Medicine and Neurology at the Edgware General Hospital. This post was followed by four further posts of increasing seniority at St Mary's Hospital, culminating in the appointment to Research Associate in the Waller Department of Cardiology in 1978. Russell is also an Honorary Research Fellow at the London School of Hygiene and Tropical Medicine; a Clinical Assistant in the St Mary's Group of Teaching Hospitals; and a Principal in General Practice in Hertfordshire. He achieved his MRCGP from the Royal College of General Practitioners in 1984.

15 Phonetics training on the IBM PC

ANTHONY BLADON

*Phonetics Laboratory
University of Oxford
41 Wellington Square
Oxford OX1 2JF*

ju:nivɜ:səti levl stju:dnts əv læŋgwɪdʒɪz əv spi:tʃ pəθvələdʒɪ ænd əv
lɪŋgwɪstɪks ju:ʒvəli ni:d tə lɜ:n haʊ tə trænskraɪb saʊnz fənetɪklɪ laɪk
ðɪs

Introduction

University-level students of languages, of speech pathology, and of linguistics, usually need to learn how to transcribe sounds phonetically, like the above. A phonetic transcription is called for, wherever there is need to make an accurate record of pronunciation, whether because the language itself does not reflect pronunciation very directly in its spelling (as in English), or because the speaker uses an unexpected pronunciation (such as a regional accent, or disordered speech), or simply to assist with good foreign language learning. For the instructor however, the training of students in phonetic transcription is largely a routine and repetitive task. It is a task which lends itself to automation for this reason and a number of others:

- (1) students need continuous feedback;
- (2) it demands meticulous assessment of fine detail;
- (3) model answers can reasonably be defined;
- (4) it lends itself to quantitative assessment.

We therefore thought it worthwhile to develop, design, and test a computational alternative. CAPTEX (Computer Assisted Phonetic

Transcription Exercises) is the result. CAPTEX, as currently implemented, runs on the IBM PC range, in Basic (IBM Basica). There are no special requirements other than a colour graphics screen and card, both of which are necessary in order to display phonetic symbols. The CAPTEX program is loaded automatically once the student user inserts a floppy disk; the student begins working on a transcription exercise directly, assisted by menus, prompts, and instructions on the screen. CAPTEX primarily does the following:

- (1) presents the student with ordinary orthographic text;
- (2) provides for the typing in of a phonetic transcription of that text;
- (3) scores the transcription effort;
- (4) allows retries in the case of an error;
- (5) and meanwhile supplies interactive help or comments on the nature of the error or how to correct it.

At the point where the student concludes the exercise, CAPTEX displays a summary of its assessment of the work session. Students can obtain a paper printout of whatever is on the screen, which might therefore show their work as it progresses during the exercise itself, or the end of exercise assessment. Throughout the work session, and hidden from the student's awareness, the IBM PC keeps a log of activity, which the instructor can later inspect in order to determine who has achieved what, and when.

At the same time, the CAPTEX software allows for further development. Additional programs permit the following:

- (1) the adding of new exercise material;
- (2) the adding of new phonetic symbols;
- (3) changing the keyboard location of the phonetic symbols;
- (4) changes to the comments and guidance which the computer supplies in response to a transcription effort;
- (5) adding additional alternative transcriptions of a word which the computer will judge as 'correct' (or, as 'incorrect').

These are regarded as key features of the training software. It is most important to allow the instructor some flexibility on a number of issues. For example, provision must be made for different variants of the chosen transcription system, depending on the depth

of transcriptional detail one wishes to train (stress marks, details of aspiration, fine vowel quality, etc.), depending also on the style of speech assumed, and indeed even the accent to be transcribed.

Some of the capabilities of CAPTEX can be appreciated from a closer look at a training session. A view of the computer screen at a typical point in a student's exercise is shown in Fig. 15.1. The top three lines contain messages and prompts; the phonetic transcription typed in by the student appears immediately underneath the ordinary orthographic text (which is supplied by the computer). Phonetic symbols which are not part of the Roman alphabet are typed as combination keystrokes, making use of the high valued ASCII numbers in the 128-255 range (the IBM 'Additional Character Set', redefined). Thus, for example, holding down the IBM 'Alt' key and typing 's' will yield the phonetic character 'ʃ'. We are very grateful to IBM United Kingdom Scientific Centre, and to David Sinclair in particular, for assistance with the various phonetics character routines. On the question of the key assignments of phonetic symbols, we have followed as closely as possible the emerging standard as presented in Wells (1986). At any point in the exercise, students may ask to have their work assessed. Feedback is therefore immediate, and tireless.

A student's effort for each word is scored as either 'correct' or 'incorrect'. This is done by means of a string comparison algorithm (due to Ceri Carlill) which determines whether a match has been obtained, and if not, to which 'correct' alternative the student's version best approximates. This two-way scoring, supplemented by extensive interactive comment and help, is reasonably suited to phonetics transcription training, but it does impose a number of constraints. The judge of what will be scored as 'correct' is of course the instructor, who has to prepare all possible alternatives which should count as 'correct', along with any comments or help associated with them. In certain cases, inevitably, the instructor's decisions are excessively polarized, and an improved version of the software might well permit a range of intermediate verdicts. Another limitation of the current assessment routine is that it does not make clear when multiple errors occur within a single word, or allow for other factors contributing to error gravity.

The English version of CAPTEX trains the system of transcription used in the major phonetics reference books and in dictionaries such as the *Oxford Dictionary of Current English* and the *Penguin English Dictionary*. In certain areas, the software has

[tu:] is incorrect.
 Use a weak form (weak vowel) normally.
 Enter your choice: F1 - RETRY transcription; F2 - Get ANSWER:

We can turn now to the development of language,
 wɪ kən tɜ:n naʊ tu: ðə di'veləpmənt əv læŋgwidʒ

Learning a native language is an accomplishment within
 lɜ:nɪŋ ə neɪtɪv læŋgwidʒ ɪz ən əkəm'plɪʃmənt? wɪðɪn

the grasp of any toddler, yet discovering how
 ðə grɑ:sp əv enɪ tɒdlə je? dɪskʌ'verɪŋ haʊ

children do it has eluded generations of philosophers
 tʃɪldrən du ɪt hæz ɪlu:did ↑

and linguists. Saint Augustine believed it was simple.

User: BLADON Exercise: LANGUAGE

Fig. 15.1 IBM PC screen display at a typical point in the phonetics training exercise. The student has transcribed phonetically as far as the word 'generations' (see cursor), and has then asked to have the work assessed. Assessment has proceeded as far as the word 'to' (see underline, and messages at top of screen).

incorporated systematic decisions on pronunciations which are currently undergoing change or variation. These range over such matters as the use of vowel reductions in unstressed function words (*can*, *at*, etc.), glottal stop for [t] before a consonant, [ɪ] versus [ə] in unstressed affixes, *when* pronounced with a voiced initial consonant (i.e. not as 'hwen'), *more* pronounced with no diphthongal glide, syllabic consonants, and reduced triphthongs. Decisions on these matters reflect recent phonetic research, and any interested reader may obtain full details from the author.

Used in the way described, the CAPTEX program automates only that kind of transcription exercise in which the material to be transcribed phonetically is in text form. Transcribing from *spoken* input, for example, is not yet possible.

This is obviously desirable for more advanced students especially, since it permits a realistic control over the transcription of accent differences, speech rate and style differences, and prosodic features. To look forward to an automated version of transcription from speech is not too futuristic a prospect. It is quite feasible to integrate the students' playback of previously recorded speech on cassette, in a manner which ties in with the screen text and prompts. More ambitiously, the IBM PC can already be fitted with a speech-synthesizer board which generates speech directly from any text input whatever (and this could likewise be tied in with the screen activity). But despite the continued advances in quality of such synthetic speech, no synthesizer is yet known (at the time of writing) which would achieve an acceptable mimicry of natural speech to serve our purposes. Once this quality is available, however, the synthesizer becomes a really attractive option, because it allows for controlled manipulation of whatever speech parameter (say, details of vowel quality or pitch-range variations) one may wish to train.

Reference

- Wells, J.C. (1986). *A standardised machine-readable phonetic notation*. IEE International Conference on Speech Input/Output; Techniques & Applications. IEE Conference Publication No. 258, pp. 134-7.



ANTHONY BLADON

Anthony Bladon is university lecturer in phonetics and Fellow of Wolfson College, Oxford. Having started out in modern languages at Cambridge, then in linguistics at Reading and Bangor he is now, like most of the phonetics community, very much involved with research applied to the speech technology industry: computer speech synthesis and automatic speech recognition. As Chairman of

the management committee of the Computing Teaching Centre of Oxford, he is further able to combine these interests at a teaching level (as his contribution here shows).

16 Thermoluminescence dating: an archaeological application of the IBM PC

I. K. BAILIFF

*TL Laboratory
Department of Archaeology
Durham University
Fulling Mill
The Banks
Durham, DH1 3EB*

Archaeology is a discipline with rapidly expanding frontiers, and in recent years much of its expansion has been concerned with the application of scientific methods, such as dating.

Thermoluminescence (TL) dating is an absolute dating technique which is commonly applied to inorganic artefacts such as pottery (whereas the dating of organic remains is the province of carbon 14 dating). TL dating is a relative newcomer to absolute dating studies and, it too, is undergoing expansion in diverse areas of application. In addition to the dating of archaeological pottery, the research field embraces areas such as the authentication of ceramic works of art, the dating of geological sediments, and the measurement of atomic bomb doses.

The phenomenon of thermoluminescence, as the name suggests, is associated with *heat* and *light*. It is a property, commonly found in crystals, that enables a minute fraction of the energy available in an ionizing radiation field to be stored and later released in the form of light upon the action of heating. That is, if a radiation dose is administered to TL crystals and the crystals are subsequently heated (usually rapidly at a constant rate), weak light emission, referred to as TL, may be observed. The emission occurs in particular temperature regions, giving rise to a *glow curve* (see Fig. 16.1), the structure of which is characteristic of the type of TL mineral and

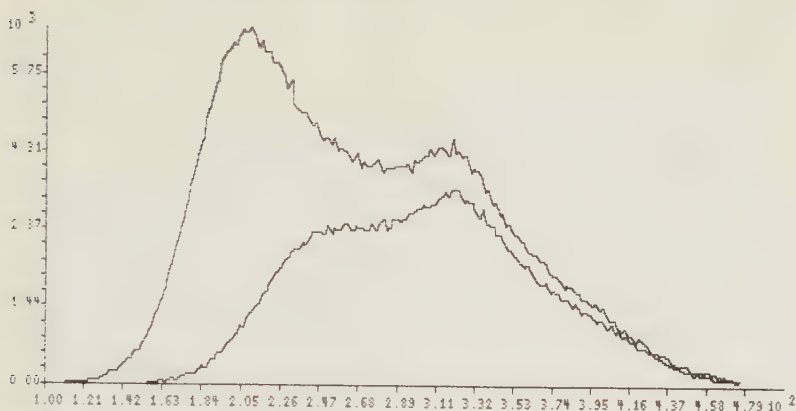


Fig. 16.1 The TL signal (ordinate) plotted as a function of temperature (abscissa) in degrees Celsius, producing a TL glow curve. The TL glow curves show: (a) the curve obtained on first heating (from 20 to 475 °C quartz crystals extracted from Iron Age pottery, and (b) after subsequent administration of a known dose of ionizing beta radiation to the same portion. The heating rate was $10^{\circ} \text{ sec}^{-1}$ and the signal due to thermal radiation from the heater plate has been subtracted.

reflects the nature of the electron and hole-trapping centres that are connected with the TL process. The latent TL information can be stable for periods in excess of a million years, depending on crystal type and temperature region of the TL glow curve. The TL phenomenon can be used as the basis of a dating technique for pottery because pottery contains two essential ingredients: TL minerals and naturally occurring radioactive impurities which give rise to a weak ionizing radiation field within clay (and in the soil when buried).

Firing of the pot sets to zero all the previously acquired latent TL and, since the radioactive impurities in the clay are unchanged by firing, the latent TL builds up with time. In the laboratory TL crystals are extracted from the pottery and heated to measure the TL signal. The signal recorded represents the total accrued radiation dose delivered to the crystals since firing, and this can be evaluated by measurement of the response of the crystal samples to known doses administered in the laboratory. Other measurements are also performed on the pottery fabric and the burial soil to determine their natural radioactivity, from which the radiation dose-rate to the crystals is calculated. The TL age is simply the ratio of the accrued dose and the dose-rate per annum.

Apparatus

In conventional TL dating apparatus a photomultiplier is used to detect TL emission from the sample, which is located on a resistive heater plate. The photomultiplier is operated in pulse-counting mode and a photon ratemeter performs the appropriate conversion of pulse to voltage signal. A thermocouple, fastened to the underside of the heater plate, is connected to an amplifier/cold junction circuit. The TL glow curve is obtained by feeding the analogue voltage outputs of the photon ratemeter and the thermocouple circuit to each axis of an X-Y recorder.

At the first stage of the development of our apparatus we have used the IBM PC, in conjunction with a 68000-based interface system designed by the Microprocessor Centre of the University, to perform data acquisition and processing for TL glow curves. The conversion of the TL data to digital form considerably simplifies examination and comparison of the large number of glow curves necessary in the dating experiments, which were previously analysed by hand.

Operation

The interface is designed to record, at selected time intervals (typically 200 ms), the temperature signal from the thermocouple and the accrued photon count while the sample is heated. Parameters for the interface control program (including start and stop temperatures, and count interval), are entered on the IBM PC and the program is downloaded to the interface via an RS232 port (when all development has been completed the program will reside in EPROM). The heating cycle is commenced manually and, once the start temperature has been exceeded, the counter and the thermocouple ADC (analogue to digital converter) are read after each time interval, and their values displayed as a pixel within a pre-defined screen area on the IBM PC. The counter is then reset and the process continues until the stop temperature is achieved, thus providing a trace representing TL intensity vs. temperature on the IBM PC display (a standard colour monitor operating in monochrome mode has sufficient resolution). On termination of the heating cycle, a file containing the TL/temperature/parameter data is automatically written to either floppy or hard disk. Typical hard copies of recorded glow curves are shown in Fig. 16.1 and 16.2. The

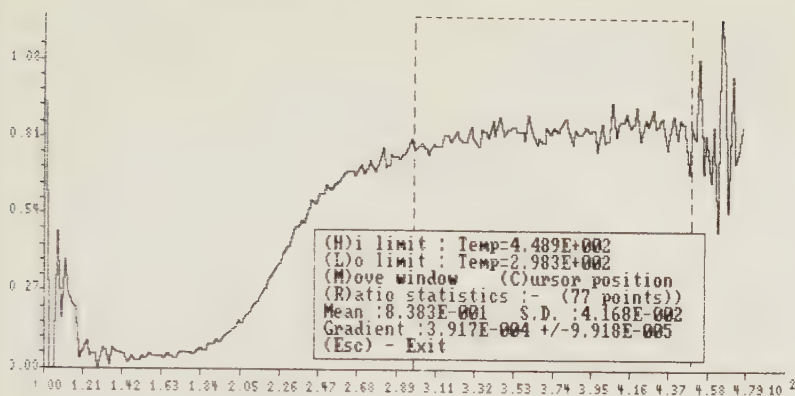


Fig. 16.2 The ratio (ordinate) of curves (a) and (b) in Fig. 16.1, plotted as a function of temperature (abscissa). A similar test is performed in dating measurements to test for stability of TL during antiquity.

curves shown were obtained from a relatively bright sample, but in many cases the signal is considerably weaker. For such cases, and also to simplify the analysis of TL glow curves, we have developed programs for Fourier analysis of glow curves (see Fig. 16.3). The installation of an 8087 coprocessor has substantially reduced the time required for such floating-point intensive calculations. Programming for the various curve analysis operations was initially in Fortran for computation and MS Quick Basic for graphical

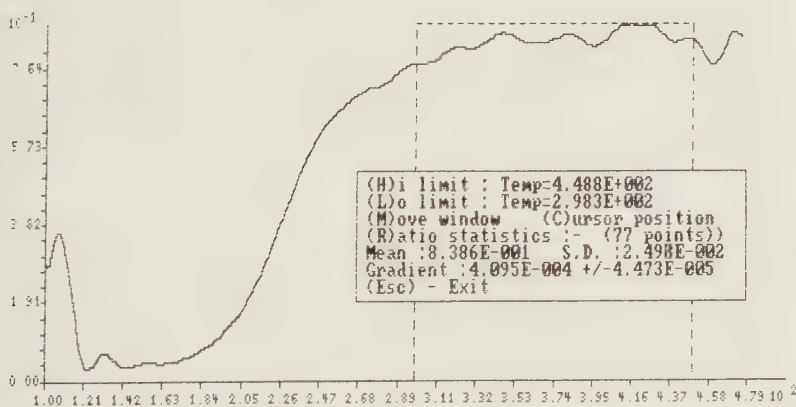


Fig. 16.3 The ratio, as shown in Fig. 16.2, but using Fast Fourier Transform reconstructions (25 terms) of the TL glow curves.

display (this is partly historical since early work on the interface used GW Basic). However, we have been much impressed with the performance and flexibility of Borland's Turbo Pascal and development is now proceeding with this compiler family.

For each sample, the measurement of at least 30 TL glow curves is required to evaluate the accrued dose (we can expect samples to exhibit TL, but their characteristics vary significantly). A number of techniques have been developed to evaluate this accrued dose; all yield a series of glow curve data obtained under various measurement conditions. These curves need to be examined using standard checks, and then compared, in order to obtain a plot of TL signal vs. laboratory radiation dose, referred to as a 'growth characteristic', from which the accrued dose is evaluated. A series of programs have been developed which perform:

- (1) the location of TL peaks and, using these, the relative adjustment of glow curves along the temperature axis to correct for small differences in thermal conductivity between the sample disc and heater plate;
- (2) the comparison of TL intensities obtained at temperature points within a selected interval for a series of glow curves (see Fig. 16.2); and
- (3) the plotting of the growth characteristic for selected temperature points, in most cases using linear regression.

In evaluating the accrued dose from the latter, the ideal sample exhibits constancy in accrued dose estimate with temperature; for example, evaluations are commonly made at two-degree intervals between 300 and 400 °C for archaeological pottery samples. After input of the dose-rate data, the TL date is calculated, together with an assessment of random and systematic errors.

Automation

Despite the use of microprocessor-controlled systems to digitize and analyse TL data, much bench time is required by the operator to analyse samples. For this reason we are at an advanced stage in the development of an automated TL reader that will perform the necessary measurements for 24 sample portions (Fig. 16.4). Experimental sequence control is achieved using the 68000-based interface, and the control program parameters (for operations such

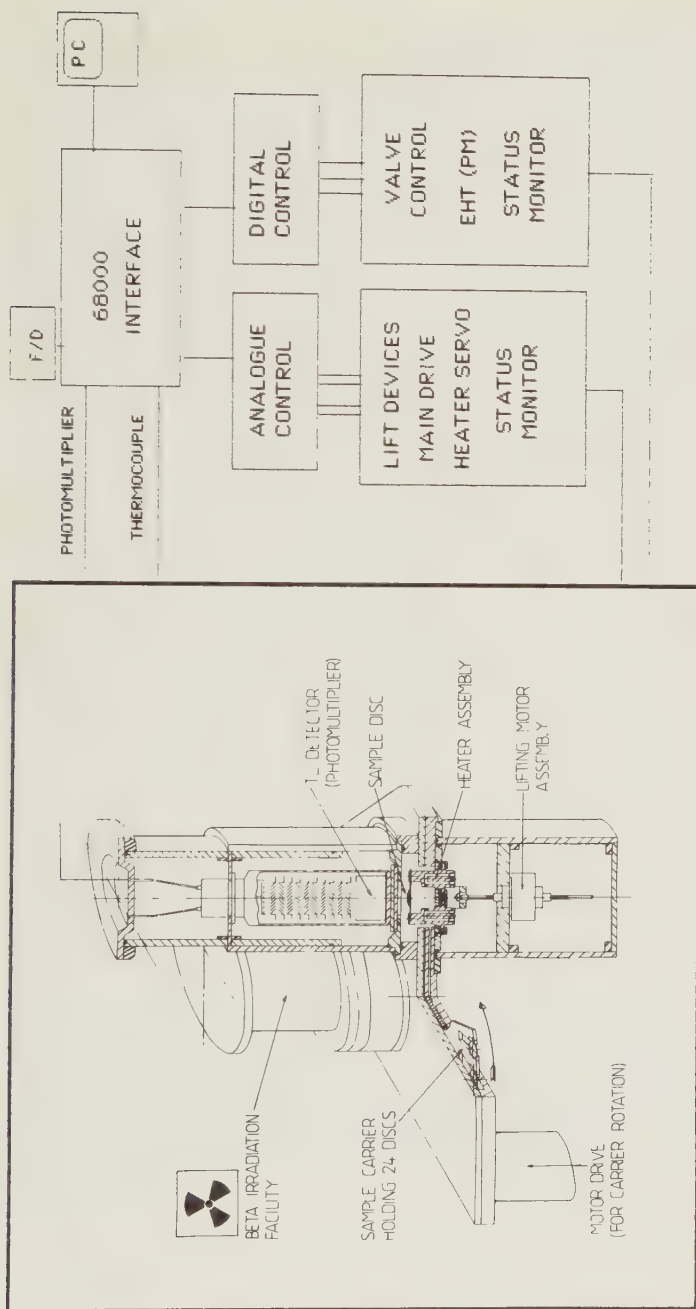


Fig. 16.4 A schematic diagram of the automated TL reader, showing the experimental apparatus linked to the 68000 interface. The 24-sample carrier disc is rotated using a closed-loop servo-motor drive system, and two lift devices raise a selected sample into either a TL measurement or an irradiation position. The carrier chamber is evacuable. Switching of various valves, and control of analogue devices, including monitoring of status, is performed by the 68000 via a Control Interface (copyright: TL Laboratory, Durham University).

as TL measurement or beta irradiation of each portion) are presently set using a simple command system entered with a text editor on the IBM PC before a run commences. Once the parameters have been downloaded to the interface, the IBM PC is free for analysis of previously acquired data that has been written to the interface floppy disk and then transferred to hard disk. Fourier analysis of glow curves enables considerable compaction of data and, although the programming of this system is not yet complete, we hope to be able to make use of Prolog to assist in the management of experiments.

TL dating service for archaeologists

The IBM PC also performs a valuable function as an administrative tool for the Durham TL Dating Service, which is a unique facility used by archaeologists from Britain and abroad. We have recently dated a piece of pottery recovered from the excavation of a spectacular logboat found in the Humberside area (Fig. 16.5) that supports its Iron Age origin. The laboratory has also been participating in the Joint US/Japan reassessment of dose delivered at Hiroshima and Nagasaki; a sensitive TL technique, originally developed for dating medieval and younger ceramics has been successfully applied to measure the bomb radiation dose that had been delivered to bricks and tiles from the two areas.

New dating research

Our research has recently expanded into a new area where TL has great potential. In collaboration with colleagues in the Department of Geography, whose research is concerned with sea-level changes during the past 10 000 years (see Fig. 16.6, noting the rise in excess of 20 m), we are investigating the possibility of dating sediments from coastal regions within this timespan. TL offers a means of directly dating the mineral sedimentary deposits that result from the action of the sea, whereas the previously used method of carbon-14 dating has presented problems because of the lack of availability of suitable organic material. The measurement of sea-level change, and its placement on an absolute timescale, is of importance because of the implications for future land-use developments in coastal lowlands, e.g. power stations. Ultimately we hope that this work will lead to archaeological applications in dating deposits from Mesolithic, and earlier, sites.



Fig. 16.5 A sherd, recovered from the excavation of the Hasholme logboat was dated by TL to AD 190 ± 270 years using the Service's low accuracy Survey dating. In a detailed dating program an accuracy of between ± 5 -10 per cent of the TL age can be expected. The Hasholme logboat was excavated under the direction of Dr M. Millett of this department. [Photograph by M. Millett; copyright: Hull Museums.]

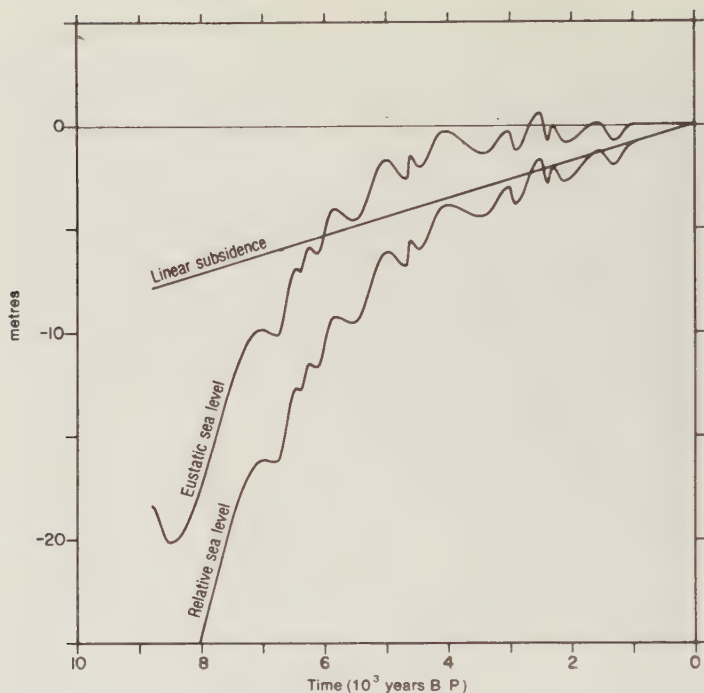


Fig. 16.6 A plot of eustatic sea-level change, i.e. a change not arising from an alteration in the elevation of the land, that, in combination with linear subsidence, results in the relative sea-level change (shown by the lower curve) for the UK fenland over the last 10 000 years (copyright: I. Shennan, University of Durham).

Conclusion

In this broad sweep of TL dating and the use of the IBM PC, the flexibility of its role in the laboratory is a notable feature. The potential of the IBM PC to assist in other areas of research, such as aerial photography and the formation of regional archaeological databases, is now being explored.

Acknowledgements

The TL Laboratory is pleased to acknowledge grant support for work described in this paper from the Nuffield Foundation, the Science and Engineering Research Council, the Manpower Services

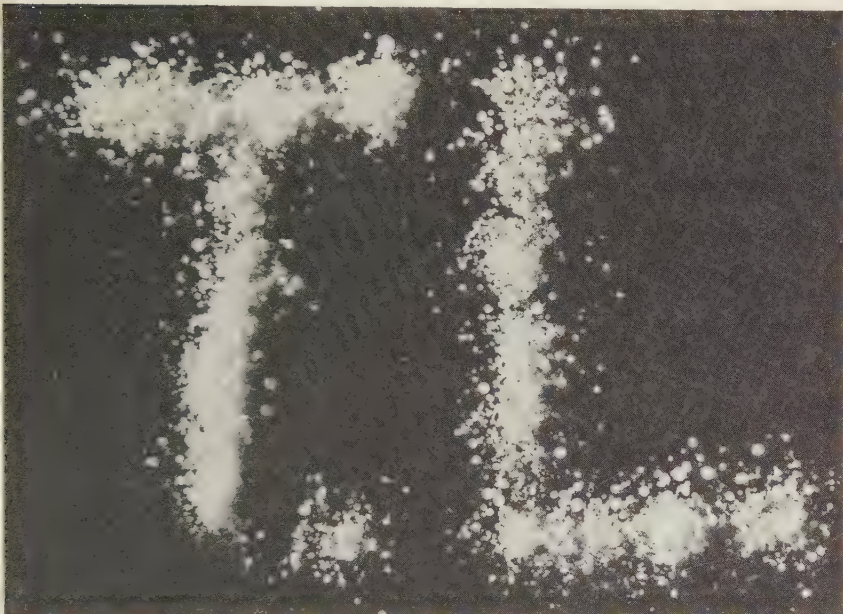


Fig. 16.7 Simulated blue TL emission from crystals (copyright Archaeology Department, Durham University).

Commission, IBM United Kingdom Ltd, and the University of Durham.

Recommended reading

Aitken, M.J. (1985). *Thermoluminescence dating*. Academic Press. London.

Ancient TL, a newsletter for the TL research community edited by the author.



IAN BAILIFF

Ian Bailiff is currently SERC Advanced Fellow in the Department of Archaeology at Durham University. He came to Durham in 1978 via Sussex and Oxford Universities after reading physics and then specializing in TL applied to dating. He has developed a TL dating laboratory, the aim of which has been to further the integration of TL dating studies in archaeology. He has participated as consultant in the joint USA/Japan reassessment of bomb dosimetry at Hiroshima and Nagasaki, and is editor of *Ancient TL*.

17 Sheep health, productivity, and the IBM PC

K. L. MORGAN

Department of Veterinary Medicine

and A. T. A TUPPEN

The Computer Centre

University of Bristol

Langford House

Langford

Bristol, BS18 7DU

Introduction

The Welsh Mountain and Scottish Blackface sheep originate from the two most densely populated sheep farming areas in the world. They are but two of the 40 breeds and 35 million sheep which make Britain the largest producer in the European Economic Community.

The health, welfare, and productivity, of the national flock depend on the skill and care of individual shepherds. Traditionally, the contribution of veterinarians has been mainly confined to specialized obstetrical procedures around lambing time.

Recently, there has been a change in veterinary teaching, with more attention being paid to the care of the flock throughout the year and a change of emphasis to prevention rather than cure. Preventive measures are not new to sheep farmers, as vaccination, dipping, and foot care are well-established routines of sheep husbandry. However, there are a number of other preventive procedures which may be instituted on a farm to combat specific problems, and this requires a knowledge of the type and extent of the problem, and needs a method of recording.

Some sheep farmers already keep good records. In small pedigree flocks, information pencilled into a well-thumbed notebook records details of genealogy, prolificacy, mortality and, in some cases, disease. Unfortunately, little use is made of this information because

retrieval is difficult and time consuming. This system of recording becomes more difficult as the flock size increases, and on flocks of 1000–2000 sheep, farmers may be unaware of the number and extent of health and productivity within their flocks.

The system

During the past 18 months we have developed a database to record productivity and disease in commercial sheep flocks. We began writing the package using dBASE II on an eight-bit microcomputer with a 10 Mbyte hard disk. It soon became apparent that this combination was unsatisfactory as it was too slow, and lacked the mathematical functions necessary for statistical analysis of the recorded information. So, we changed to the much-improved dBASE III and as this necessitated a 16-bit microcomputer, we chose the IBM PC XT.

We set out to develop a database using the following criteria:

- (1) it should be suitable for use in research and general practice at all levels of sophistication;
- (2) it should be able to record details of individual sheep, such as those shown in Fig. 17.1, in flocks of up to 2000 sheep over a



Fig. 17.1 Screen display for the recording of details of individual sheep.

five-year period, and with cross referencing between years and familial relatives; and

- (3) it should record details of nutrition, management, and climate, to allow for correlation of epidemiological data.

We began by making a comprehensive list of the information we wished to record, attempting in the process to predict and incorporate details which, although of little relevance at present, might be of value at a future date. An example of this is the climate database which allows for recording of temperature, rainfall, wind, and chilling on a weekly basis. Climatic factors are known to influence disease, and we would envisage epidemiological studies incorporating the use of an automatic weather station on each of the farms involved, although this is not feasible at present.

To avoid repetition, the information was classified according to whether recorded on a flock or an individual basis. In some cases this involved fine divisions, condition scoring for example. This method of determining the body condition of a sheep is independent of body size or state of pregnancy and therefore fell clearly into the sheep database, whereas the date on which condition scoring was carried out was included in the flock database. However, a certain amount of duplication was inevitable as the analysis of the recorded information sometimes required the comparison of data held in two or more databases. Duplicated information was used to link these databases together, e.g. if a field of grass had been treated with an unusual fertilizer or herbicide and we wished to find out which sheep had grazed this pasture, this would only be possible because each sheep record contains a FLOCK number linking it to the 'flocks' database, and each flock record contains a list of pasture numbers linking it to the 'pastures' database which lists the treatment of individual pastures.

This resulted in 17 databases divided between seven logical areas, with each group of databases related by common fields. The data entry and display screens were designed so that all the information in one database was displayed on the same screen. We accepted a degree of 'screen cluttering' in order to accomplish this. Nevertheless, in some instances it proved impossible; for example, the lamb and sheep databases (Fig. 17.1) were each split between three screens. In contrast, the displayed data in the climate database was sufficiently small to allow two databases to be combined on one screen.

One of the major difficulties involved the system of numbering sheep. Sheep are identified by either a tattoo number in the ear or (more commonly) by a numbered, coloured, plastic ear-tag. The tags are allocated to the database in a numerical order and are continuously recycled when the end of the set is reached. A disadvantage of this numbering system is that, although useful in individual and flock identification, it provides little information on the date of birth or pedigree of the animal. A system of bar coding could incorporate this sort of information and ease recording by speeding up data collection.

We have devoted considerable effort to data collection and entry. Having established the databases, we refined the system of data entry so that it would accept identical information for a large number of animals. It is also possible to 'customize' the data-entry screen so that a limited number of parameters are highlighted and the speed of data entry is increased. Although this system of data entry and recording is complete, we hope in future to be able to link it with a portable data recorder and an air ultrasound pregnancy scanner. Real-time scanning has now become an accepted part of sheep husbandry, allowing the identification of, and correct feed allocation to, most ewes carrying single, twin, triplet, or no lambs. Scanning is usually carried out with the ewes in a sitting position, often in a 'mobile deckchair'! In this position, the feet, udder, and teeth, may also be examined and recorded using the scanner keyboard.

The comprehensive design of the database is illustrated by the menu shown in Table 17.1. It allows for almost unlimited recording and analysis.

Table 17.1. Main menu, part 1

F	Examine the farm's database
1	Examine farm profile database
2	Examine climate database
3	Examine the fields' database
4	Examine the pastures' database
5	Examine the flocks' database
6	Examine the food database
7	Examine the treatment database
8	Examine the sheep database
9	Examine the lamb database
A	Add a new series of sheep
B	Add a new series of lambs
C	Enter identical information for a number of sheep
D	Enter identical information for a number of lambs

The menu for data retrieval is open-ended. To date we have written programs which identify the most prolific ewes, the fastest-growing lambs, and the major diseases in each flock. These identify the two major aims of this database—the productivity and health of sheep flocks. The identification and selection of ewes bearing triplet lambs and above has proved a very successful method of increasing the total number of lambs born in a flock. However, more important is the high level of perinatal mortality accepted at the moment, three to four million lambs die each year in the UK. The identification of the major diseases in the flock, and an analysis of other feeding and management techniques, will help reduce this intolerable loss.

A simple, but important part of our preventive medicine program is a calendar. Once the date of tupping—turning the rams (tups) in with the ewes has been set, a list of procedures with recommended and 'last possible' dates provides a prompt to both vet and shepherd. It also allows the dates of four or five routine visits to be arranged to coincide with the most important times of the sheep year, i.e. mid-pregnancy, just prior to lambing, mid-lambing, and weaning. The calendar also has a memo facility which lists the advice given at the last visit. In addition, it stores the written reports sent out during the year. The calendar can be updated each time it is printed, allowing the vet to review events on a particular farm prior to a visit (see Fig. 17.2).

CALENDAR FOR FLOCK 1, 1986

ME SMITH

DATE OF LAST VISIT: 01/07/86

TODAY'S DATE: 02/10/86

START OF LAMBING

Lambing starts in 85 days on the 26/12/86 (calculated from the tupping date—no raddle used)

SCANNING

scan between 15/09/86 AND 09/11/86, preferably around 10/10/86 (calculated from date of tupping)

BOOK THE SCANNER EARLY

COPPER SUPPLEMENTATION

Give copper around 15/10/86 in 13 days (calculated from date of tupping)

ORF VACCINATION

Orf vaccine must be given during the next 29 days *BEFORE* the 31/10/86 (calculated from date of tupping)

FEEDING CONCENTRATES START

Start concentrates in 29 to 43 days, between 31/10/86 and 14/11/86 (calculated from date of tupping)

DATES FOR HOUSING

House during the next 43 days before the 14/11/86 (calculated from date of tupping)

FOOT ROT (VACCINATED WITH FOOTVAX)

Vaccinate either *BEFORE* 14/11/86 or *AFTER* 09/01/87 (calculated from date of tupping)

*PASTEURELLA DATES ARE THE SAME AS THOSE FOR CLOSTRIDIAL VACCINES**CLOSTRIDIAL VACCINATION DATES*

Vaccinate between 05/12/86 and 12/12/86 (64 to 71 days calculated from date of tupping)

MEMO FROM LAST VISIT:

Condition score. Check Rams' feet. EAE vaccine. Pay bill.

Fig. 17.2 Example of the calendar.

The database has already proved useful in identifying checks in the growth rate of lambs (often the first sign of disease). It has also surprised farmers by turning the estimated prevalence of disease from 'some of our sheep get it' into 47 per cent of the flock!

An important part of the database is its facility to record 'contributing factors' in disease. The relationship between climate and the hatching time of parasite larvae deposited on grass has been well established, but the role of the environment in the development of diseases such as pasteurellosis, and the efficacy of vaccination in preventing the disease, is less clear. Our database could be used to answer these questions.

Another useful aspect of the database is its facility to identify all the familial relatives of a selected animal. This will be of use not only in genetic selection, but also in the control of diseases such as Scrapie in which genotype plays a role in susceptibility.

Development of the database will continue over the next year when it will be used on the flocks involved in Bristol University Veterinary School's flock Health Scheme. The package—called 'Bugail' (it's Welsh for shepherd and pronounced be-gile)—should be available to veterinary practitioners and farmers in 1988.



KENTON MORGAN

Kenton Morgan read veterinary medicine at Trinity College Cambridge, graduating in 1974. After a short time in practice, he joined the Animal Husbandry Department at Bristol University and completed a Ph. D. in mucosal immunology in 1979. He spent nine months as a visiting research worker at Hanover Veterinary School and three years cycling round the world, before returning to Bristol as a lecturer in farm-animal medicine in 1984.



ALAN TUPPEN

Alan Tuppen joined Bristol University as a computer operator in 1972. Since 1981 he has been a programmer in the microprocessor support unit, specializing in database programming.

18 Using the IBM PC to simulate irrigation systems

M. A. BURTON

*Institute of Irrigation Studies
Southampton University
Southampton, SO9 5NH*

Introduction

Poor management and operation of irrigation systems is currently seen as one of the major constraints to development of many irrigation schemes (Bottrall 1981; Hotes 1984). Within this discipline, management of the main system (scheduling and regulation of water supplies) has been identified as a 'blind spot' in that it has been neglected as an area of research amongst irrigation engineers (Wade and Chambers 1980).

There are numerous factors which result in poor management and operation of irrigation systems, amongst which failure to process and analyse data are significant. Processing of data in order to calculate crop water requirements is both time consuming and tedious, especially when very similar calculations have to be carried out each scheduling period (usually at either 7-, 10-, or 15-day intervals). Analysis of data is rarely done and managers are unwilling, and often unable due to lack of time, to try out different scheduling scenarios to see the consequences on scheme performance.

Microcomputers are ideal in this situation. They can perform the calculations quickly, and can act on the given database in several different ways, enabling the scheme manager to investigate various operation scenarios.

A package of programs entitled CAMSIS (Computer Aided Management and Simulation of Irrigation Systems) has been developed for use on an IBM PC XT or AT. It is intended for use on irrigation systems:

- (1) at the planning and design stage for simulation of irrigation system operation;
- (2) during the management and operation stage to process and analyse data; and
- (3) as an aid to training system managers.

The use of CAMSIS in the training mode, where the system manager is forced to make decisions and can see and evaluate the consequences of those decisions, is detailed below.

The CAMSIS package comprises an IBM PC XT with its 10 Mbyte hard disk and 384K RAM memory, a standard monochrome screen, a Pluto IBM001 high-resolution colour graphics board, and a high resolution colour monitor. A printer is also required. The software is written in Fortran 77.

Basic concepts of irrigation

In order to function properly an irrigation system requires:

- (1) the physical infrastructure to convey the water to the fields; and
- (2) an organizational structure for system management and operation.

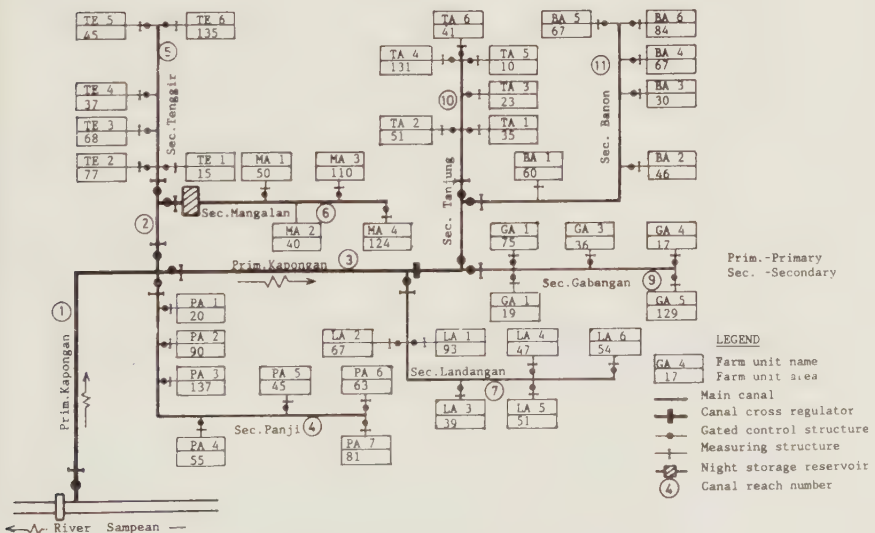


Fig. 18.1 Typical irrigation system network.

The physical infrastructure comprises a branching network of main conveyance channels together with structures at appropriate locations to regulate and measure the water. This is referred to as the main system (Fig. 18.1). Below the delivery points the water is turned out into farm channels and conveyed to the crops in the fields.

The organizational structure is generally divided into two parts:

- (1) the main system run by the government irrigation service; and
- (2) below the delivery point (the farm unit), run by a farmer, or more usually, a group of farmers.

The basic objectives of an irrigation system manager are to supply water to farm units:

- (1) in adequate quantity (discharge and duration);
- (2) at the correct time in respect of crop growth stage and condition;
- (3) with reliability;
- (4) with equity in distribution; and
- (5) with security (maintainability) of supply.

In order to satisfy these objectives the system manager must be able to:

- (1) identify and quantify the areas requiring water supplies;
- (2) identify areas requiring priority allocation of supplies;
- (3) control the water flow to suit the irrigation schedule; and
- (4) maintain water levels and discharges at or below safe levels to avoid over-topping and breaching of canals.

Within the farm unit the farmers are responsible for distribution of the water allocated to them from the main system. If crops are not to suffer water stress and hence reduction in their potential yield, the farmers must provide irrigation water in adequate quantity and at the correct time. If the water supply is insufficient for all the crop requirements, then the farmers must decide which crops and fields are to be preferentially irrigated.

Soil, crop, and rainfall data are used to estimate the crop water requirements (Doorenbos and Pruitt 1977), and river discharge data is used to calculate the water availability. From this data an irrigation schedule is drawn up detailing the discharges which are to be maintained at the control structures and to delivery points during

the next time period. This schedule is issued to irrigation department field staff who then have to regulate the canal discharges accordingly.

From these brief details the complexity of the scheduling process can be appreciated. The quantity of data that requires processing is substantial, for example a typical medium-scale system in East Java, Indonesia, has 14 main canal reaches, 85 control and measurement points, and 70 delivery (farm outlet) points, within each of which there may be 200–800 farmers' landholdings.

The main system is administered by four office staff and eight field staff, with up to 15 000 farming families dependent on the water supplies that they control. Crop data for each farm unit must be collected each time period, and rainfall, river, and canal discharges must be collected daily. There are 36 time periods in the year as a consequence of the 10-day reporting interval.

The importance of simulation in irrigation management training

In a developing country farmers may manage their plots of land for a time span of 25–30 years. In that time they will have experienced a variety of wet, dry, or average years and their irrigation water supplies may have varied not only due to vagaries of climate, but also due to such factors as canal breaches, or new or revised irrigation schedules. Also, in that time their cropping patterns will almost certainly have changed, they may be growing new crops for cash rather than food crops for subsistence.

For the farmers then, no two years will be exactly the same, they learn from experience how best to use the resources available to them and within their control. They have a reasonable opportunity, that is between 25–30 attempts, of optimizing their use of these resources. Their horizons are limited to a few plots of land and their immediate surroundings. They get to know these intimately.

The case for the irrigation scheme manager is very different. The scheme manager:

1. Will often have been transferred to the scheme from another area. He or she may not be that familiar with the locality, the people, or the scheme.
2. He or she may remain on the scheme for only a relatively short

period of time, between two and 10 years, and may be transferred just as he or she is getting to grips with the situation.

3. He or she has to deal with a complex system spread over a large area. There are many variables to cope with.
4. There will be little measure of feedback of his or her job performance, and how this may affect the overall performance of the scheme.

The last point is especially worth noting. Farmers have little difficulty in monitoring and evaluating their performances, this can be assessed from the crop yield at the end of the season. To carry out effective monitoring and evaluation of performance on a large irrigation scheme is a huge task which is rarely, if ever, done. The scheme manager thus has no feedback or indication of how his or her job performance affects the overall performance of the scheme, and the farming community within it.

Simulation can play a key role in heightening an irrigation scheme manager's awareness. Simulation offers the opportunity of trying different strategies without time-delays and without real-life consequences. The environment for simulation is non-critical, the scheme manager can try different approaches and learn from the experience without embarrassment. By working through a simulation cropping season and taking decisions on water allocations and strategies, the scheme manager will arrive at an outcome where scheme performance is measured in terms of crop yield, crop production, and cash income, both to a typical farmer and to the scheme as a whole. This simulation gives immediate feedback to the scheme manager, and provides a dramatic improvement in awareness of his or her performance.

Using the IBM PC for training in system management and operation

The CAMSIS suite of programs consists of 10 main programs for allocation of supplies on the main system (Fig. 18.2), and one program, FARMIR, for allocation of supplies within the farm unit.

For training purposes the package is organized so as to be able to:

- (1) set up a given system so that the trainee has to make decisions relating to the operation of the system for one or more seasons, (the computer carries out an assessment of the trainee's

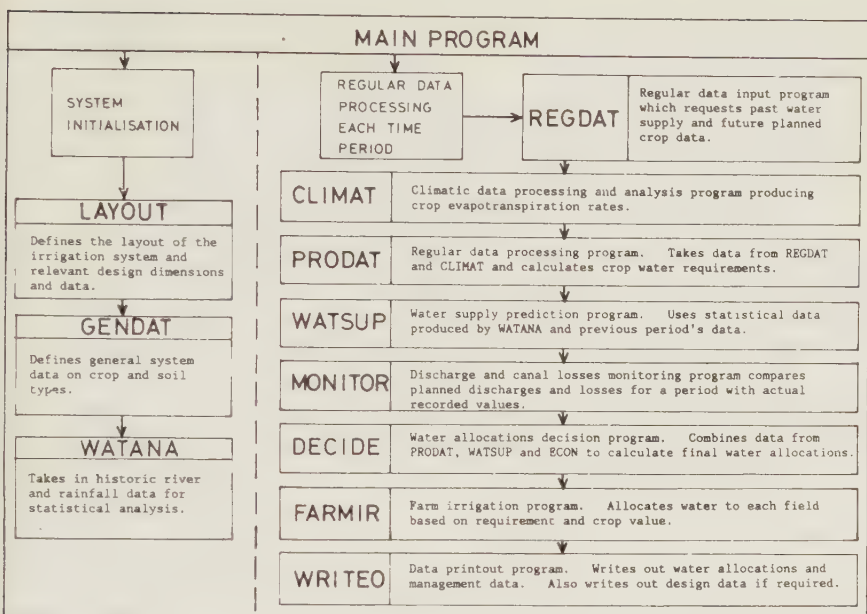


Fig. 18.2 The CAMSIS sequence of programs.

performance, and allows the trainee to analyse and evaluate the consequences of decisions taken during the season); and

- (2) set up the system so that a trainee can operate the system and become familiar with it, for example which canals feed which areas and how variations in the water supply in one canal might affect another canal.

The three programs, LAYOUT, GENDAT, and WATANA, are used to describe a particular irrigation system, while crop, canal discharge, river, and rainfall databases are created for REGDAT and WATSUP for each time period.

The trainee is presented, each time period, with data detailing what happened in the irrigation system during that time period (canal discharges, river flow, and rainfall), together with the expected cropping area for the next time period, and the expected crop water requirements. The trainee must then decide how to allocate the available water to satisfy the requirements, making the requisite decisions using the DECIDE program. The decisions are recorded, and allocations made to and within each farm unit by the

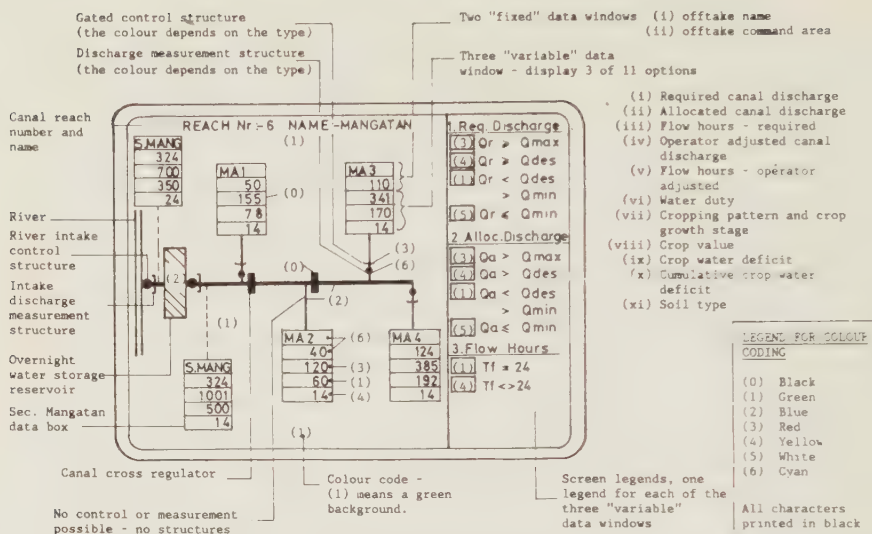


Fig. 18.3 Typical colour screen display of a canal network (from Fig. 18.1).

computer, and crop yield potentials are calculated. The cycle is then repeated with the next time period becoming the current time period.

The decisions on water allocation are made in DECIDE. Each reach of the canal system is displayed, with data which can take a variety of forms (see Fig. 18.3). Each offtake to a canal reach has a data 'box' with five 'windows'. Two of the windows have fixed data (offtake name and area supplied), and the other three windows are for variable data. Required discharge, allocated discharge, and the required flow hours are the three data variables usually displayed, though others, such as water duty ($\text{litres s}^{-1} \text{ha}^{-1}$) or crop water deficit, are possible.

The trainee states the allocated discharge for each offtake on the reach; this figure is displayed in tabular form on the monochrome screen and in the appropriate data box and window on the schematic diagram on the colour screen. Colour coding techniques are used to highlight problem situations. For instance, if a discharge in excess of the maximum design capacity is allocated to a canal, then the figure is displayed with a red background. In this instance the trainee would have to reduce the discharge to within the safe working limits of the canal (denoted by a green background). Once all the allocations are

complete, a summary printout of all the relevant management data is given by WRITEO.

At the end of one season, the total crop production and crop value for the scheme are calculated, together with leading indicators such as total water allocations to each farm unit over the season. The assessment of the trainee's performance is made against three modes of allocation adopted by the computer, and also against good results produced by previous trainees.

The MONITOR and FARMIR programs

The MONITOR and FARMIR programs are worthy of separate mention as they serve to illustrate two different yet important points.

MONITOR is used to monitor the water allocation decisions made in the previous time period and compare them with actual allocations made during that time period. The two will not always be the same. There is a random element built into the MONITOR training package which may vary the actual allocations made from the planned values. This reflects reality where control may be lost due to gate damage or a canal breach. A display of planned and

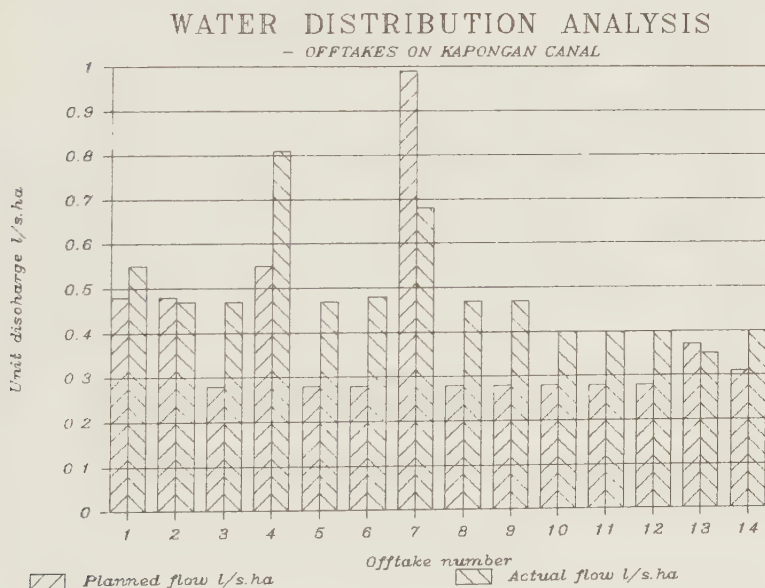


Fig. 18.4 Comparison of planned and actual unit flows to offtakes in MONITOR.

actual discharges may be given in histogram form for ease of comparison (Fig. 18.4). This program serves to emphasize the importance of the monitoring function in irrigation system management. Unlike some management enterprises, reality can often vary widely from the planned outcome.

The FARMIR program is designed to allow the scheme manager to develop an understanding and appreciation of the complexities of water distribution at the farm level. A scheme manager will rarely, if ever, need to schedule water at this level, but he or she should understand what is involved. The FARMIR program takes the water allocated each period from the main system to one delivery point (farm unit). The scheme manager then has to decide on *daily* and even *hourly* schedules for irrigating the plots of land within the farm unit. The decisions made will be based on factors such as:

- (1) interval since last irrigation;
- (2) crop type;
- (3) crop growth stage;
- (4) value of crop;
- (5) crop water demand;
- (6) water availability;
- (7) geographic location of plot within farm unit;
- (8) travel time of water to reach plot.

To carry out such an exercise on paper would take a considerable time and no doubt quickly lose participants' interest. FARMIR is structured to guide the participant through the necessary procedures, and uses the colour graphics to illustrate the situation within the farm unit. At the end of the exercise the scheme manager can thus evaluate his or her performance not only on the scheme as a whole but also at the farm unit level.

Conclusion

This paper has very briefly described part of a package of programs under development on an IBM PC XT for management and simulation of irrigation systems. When used in this manner as a training tool, such a package is extremely useful in providing an understanding of the consequences of decisions made regarding water allocations in irrigation systems. Often in real life such

understandings cannot be obtained as the feedback procedures for monitoring and evaluation are incomplete or unavailable; or else the time horizons are too long. Understanding the consequences of their actions, and being able to try different strategies for water allocations can demonstrate to system managers how significantly their actions affect the livelihood of the farming community which they serve.

The package has been developed on an IBM PC XT as it is intended that it should be used overseas on continuing irrigation schemes. For this reason a universally available microcomputer of proven ability was required.

References

- Bottrall A. (1981). *Comparative study of the management and organisation of irrigation projects*, World Bank Staff Working Paper, No. 458. World Bank, Washington, DC.
- Burton, M.A. and Farrier, D.R. (1986). *An introduction to CAMSIS—Computer Aided Management and Simulation of Irrigation Systems*. Paper presented to the Second International Conference on Simulators, Warwick University, 7–11 September. IEE Publication, No. 267, London.
- Doorenbos, J. and Pruitt, W.O. (1977). *Crop water requirements*, Irrigation and Drainage Paper, No. 24. Food and Agricultural Organisation, Rome.
- Hotes, F.L., (1984). *World Bank irrigation experience*, ODI Irrigation Management Network Paper 9d. Overseas Development Institute, London.
- Wade R. and Chambers, R. (1980). Managing the main system; canal irrigation's blind spot. *Economic and Political Weekly*, Review of Agriculture, September.

**MARTIN BURTON**

Martin Burton has recently been appointed as a lecturer in Management and Operation of Irrigation Systems at the Institute of Irrigation Studies, Southampton University, following an eighteen-month period as a research fellow developing computer programs for management and simulation of irrigation systems. Prior to working at Southampton University he worked for nine years as an irrigation and drainage engineer with Sir M. MacDonald and Partners, Consulting Civil Engineers, Cambridge. He spent the majority of his time abroad and specialized in management and operation of irrigation systems, and staff training. He is the author of several papers in these fields, and the originator of a well-used management training exercise called 'The Irrigation Management Game'.

5515 052



University of
Connecticut
Libraries

WINTER 2011

UConn

UNIVERSITY LIBRARIES

